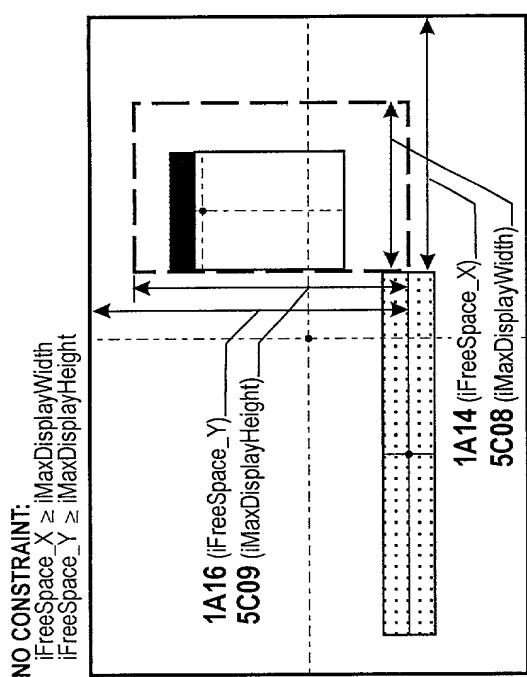
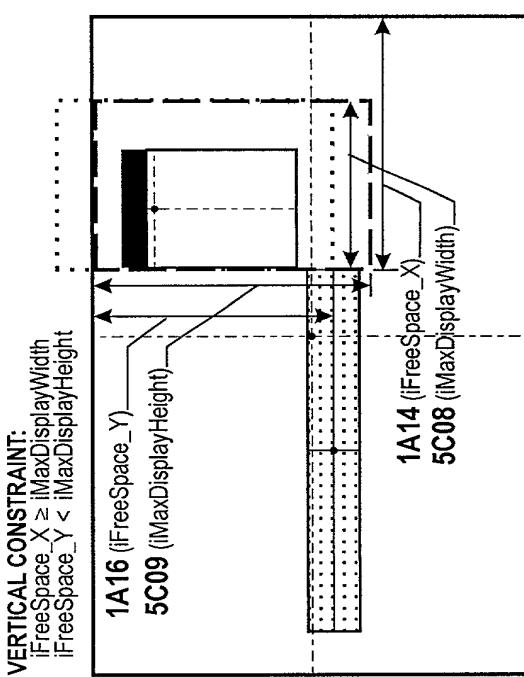
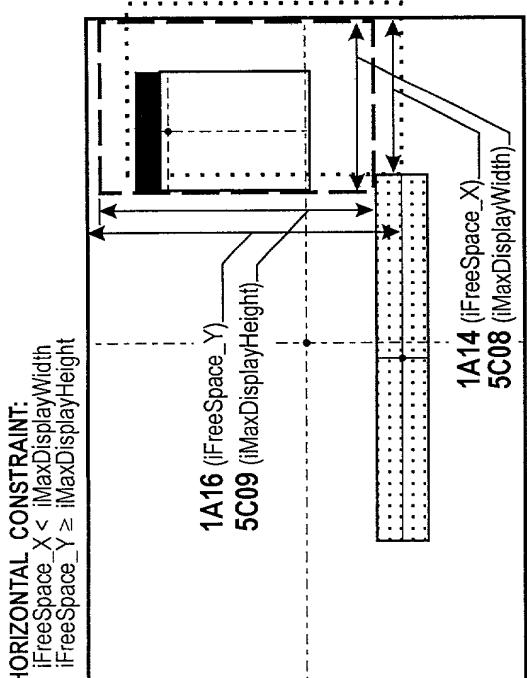
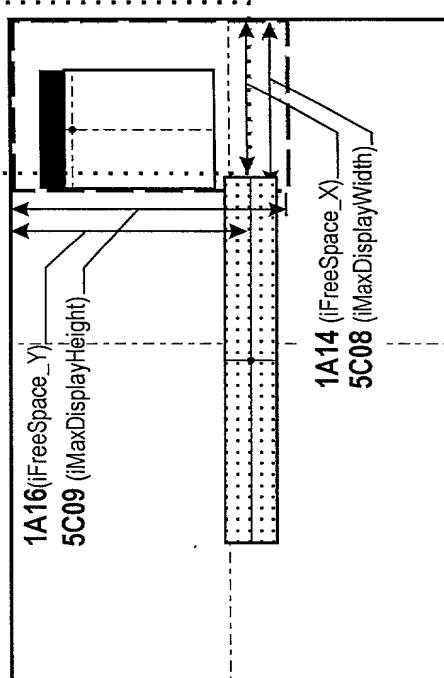
**FIGURE 1A**

**FIGURE 1B1****FIGURE 1B3****FIGURE 1B2****FIGURE 1B4**

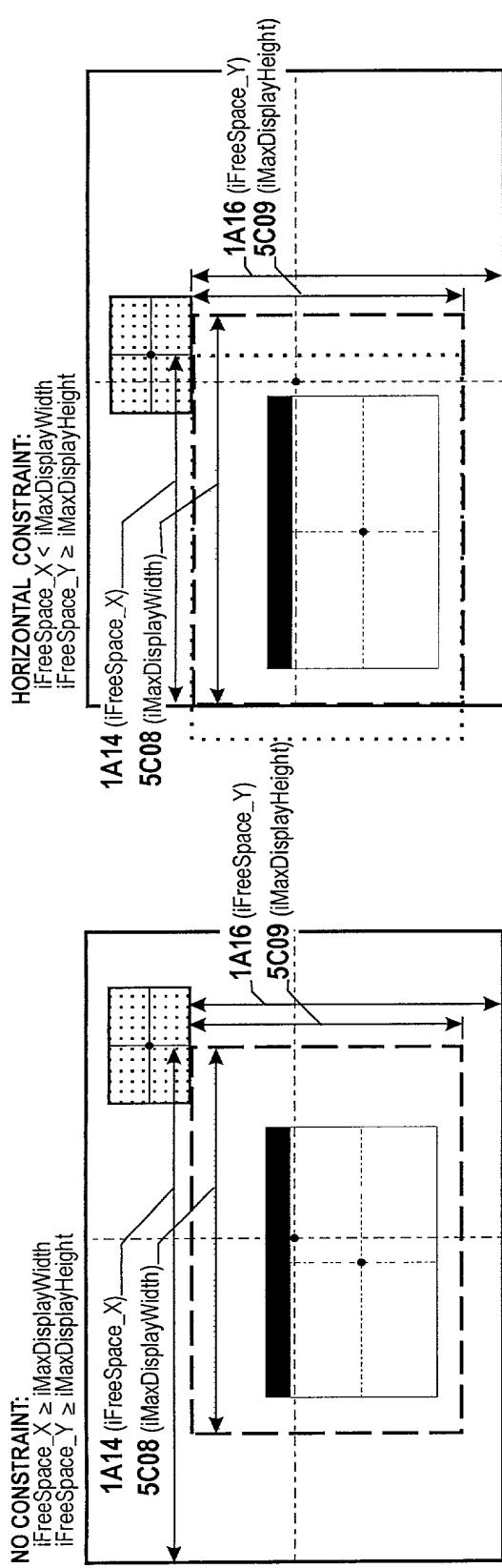


FIGURE 1C1

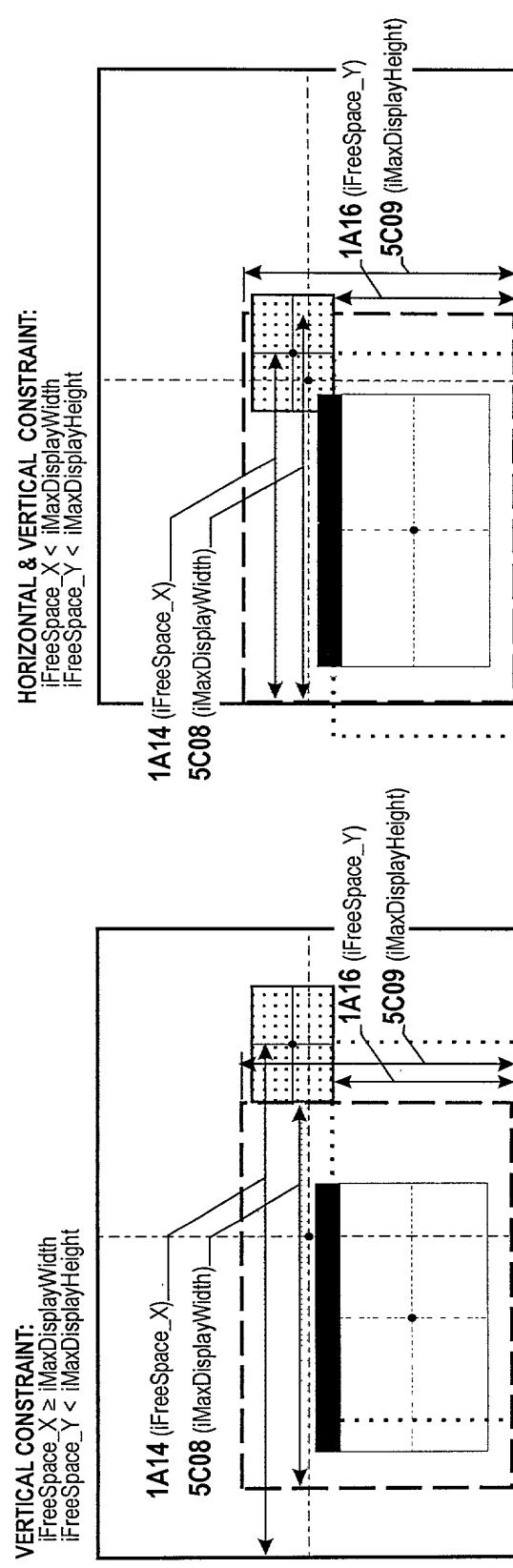


FIGURE 1C2

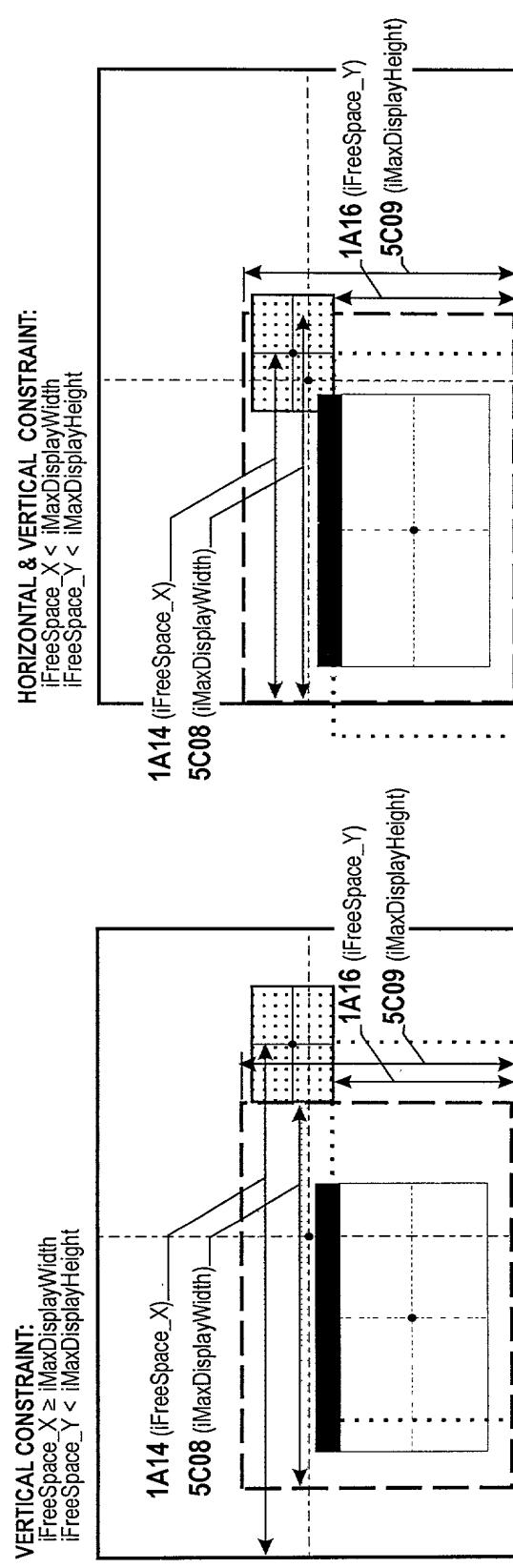
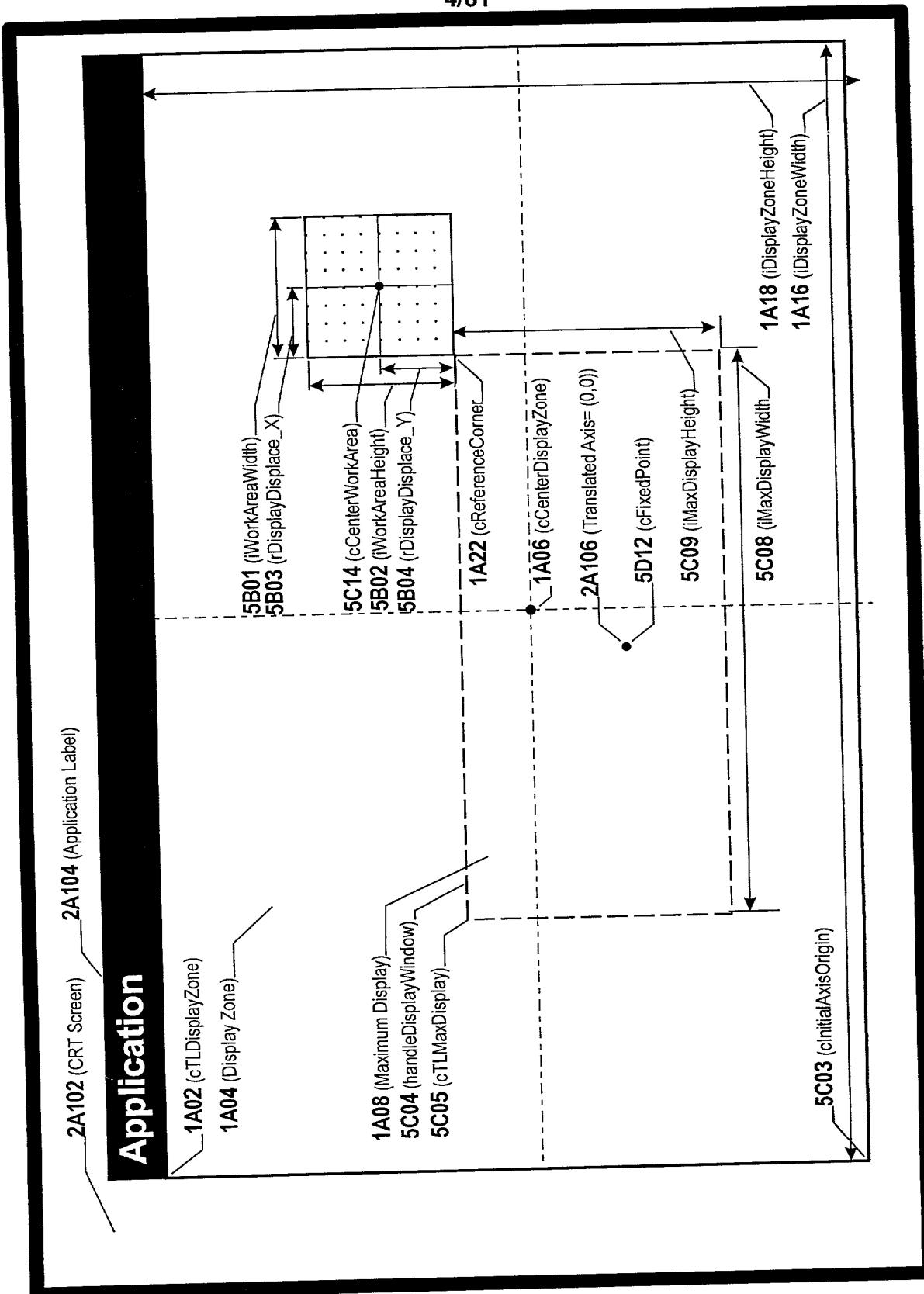


FIGURE 1C3

FIGURE 1C4

**FIGURE 2A1**

Application

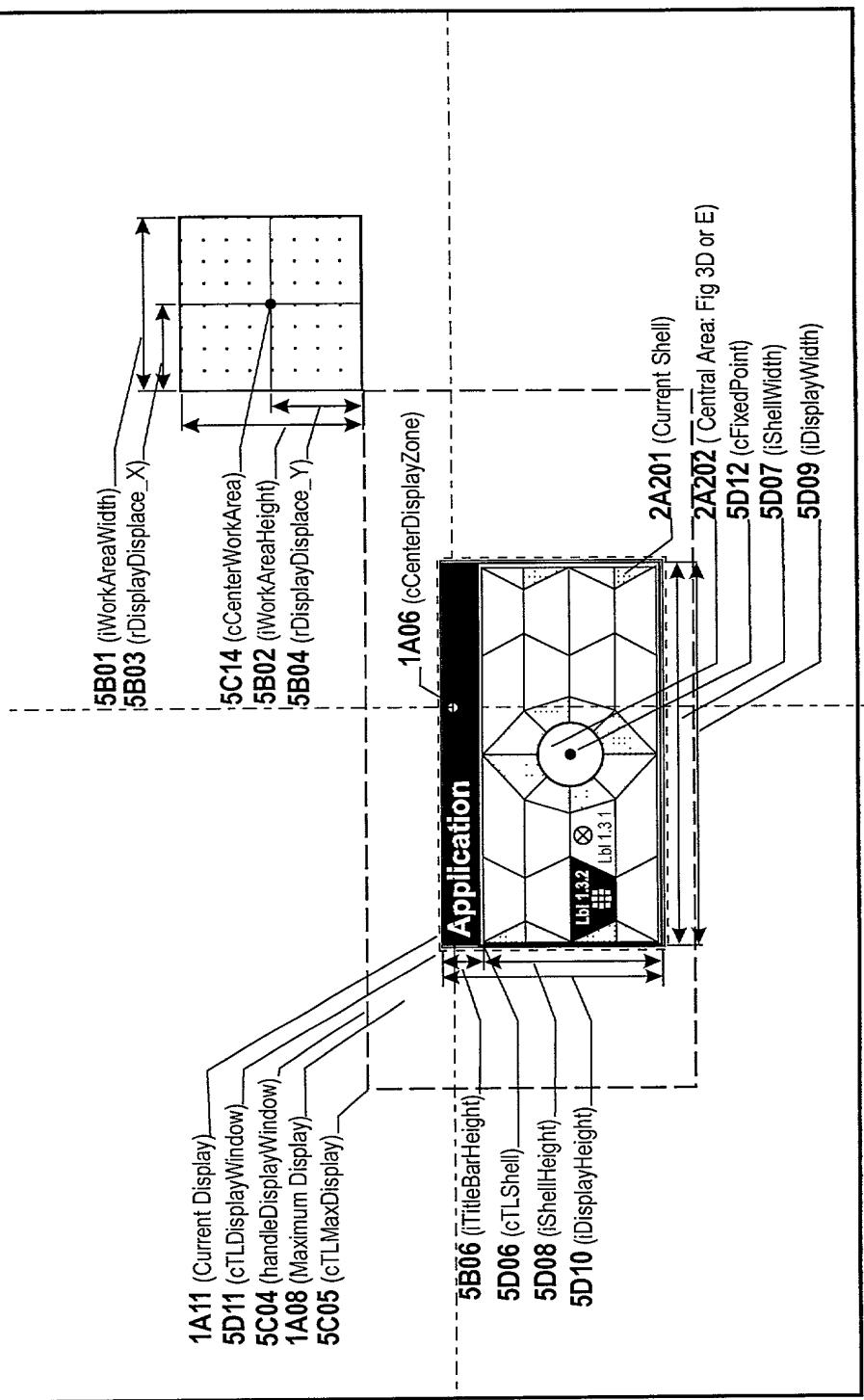


FIGURE 2A2

Application

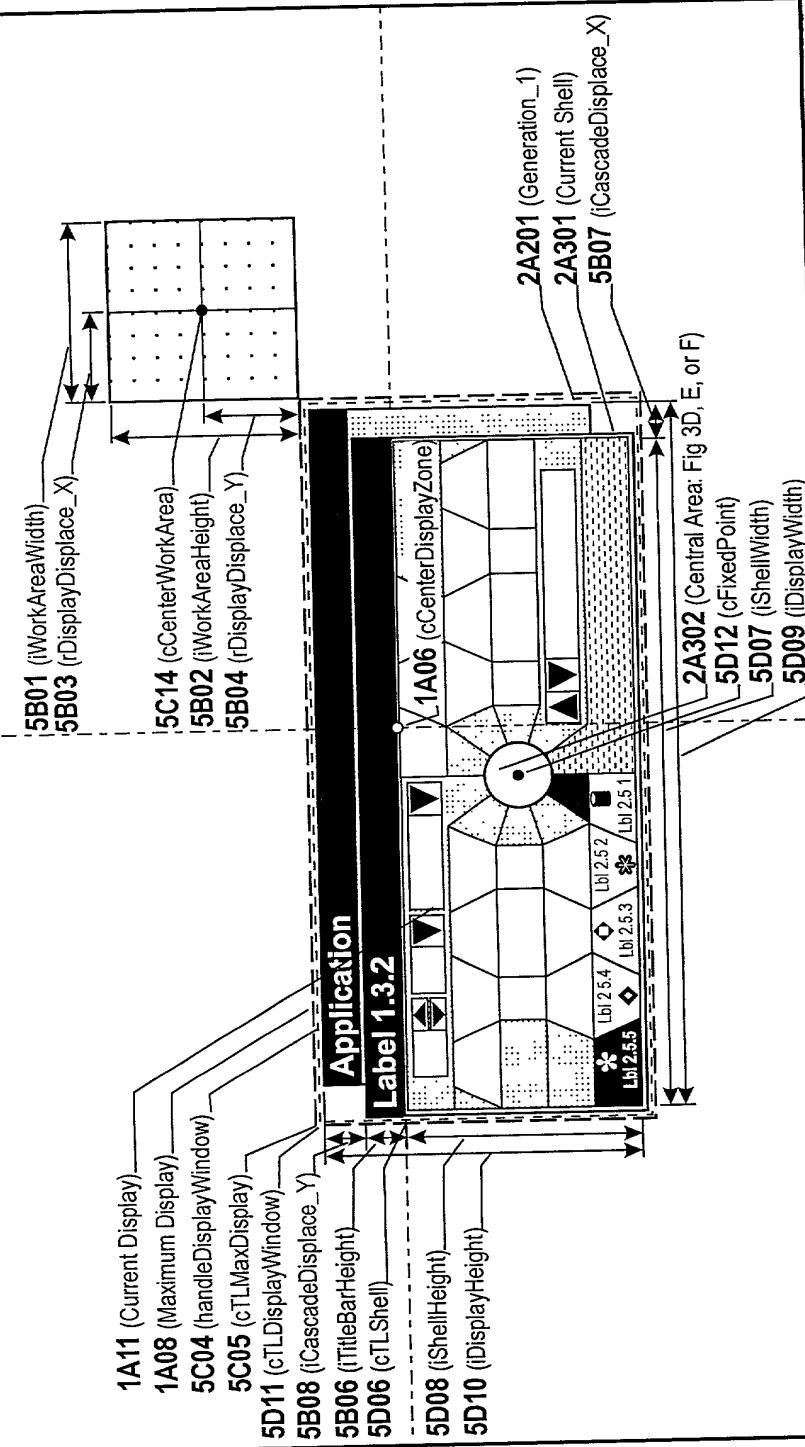
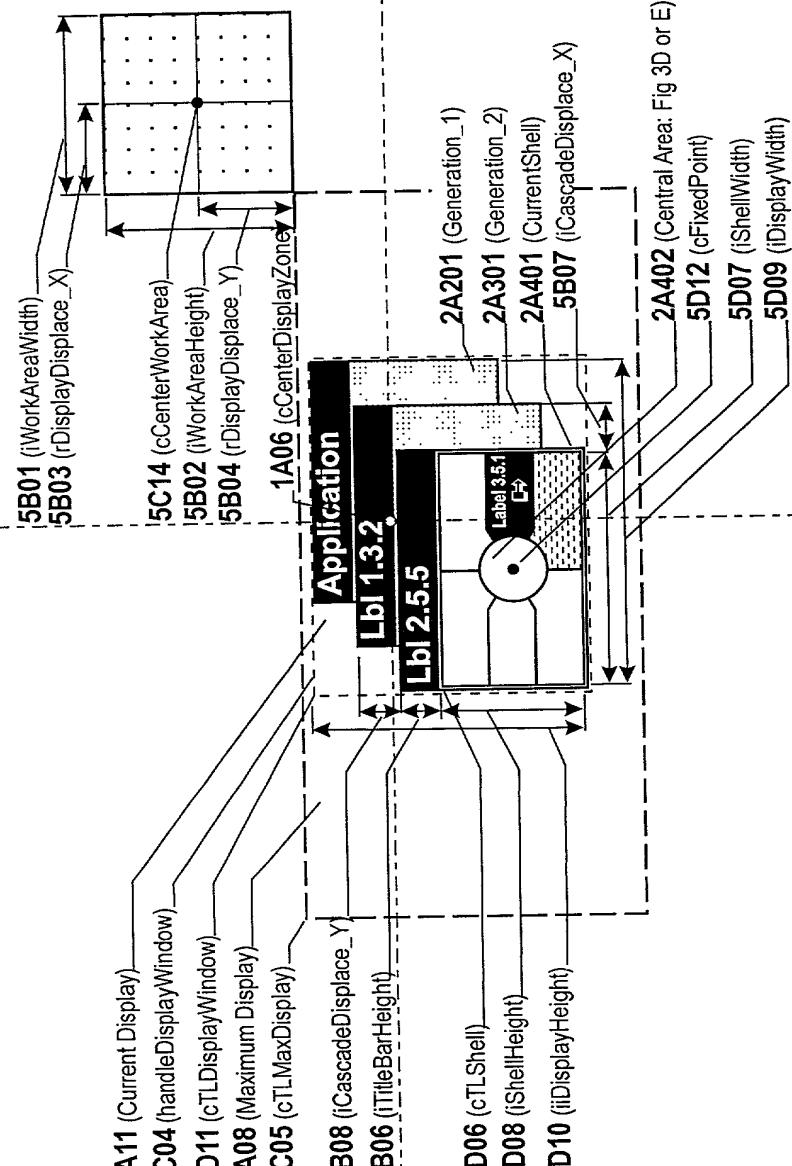
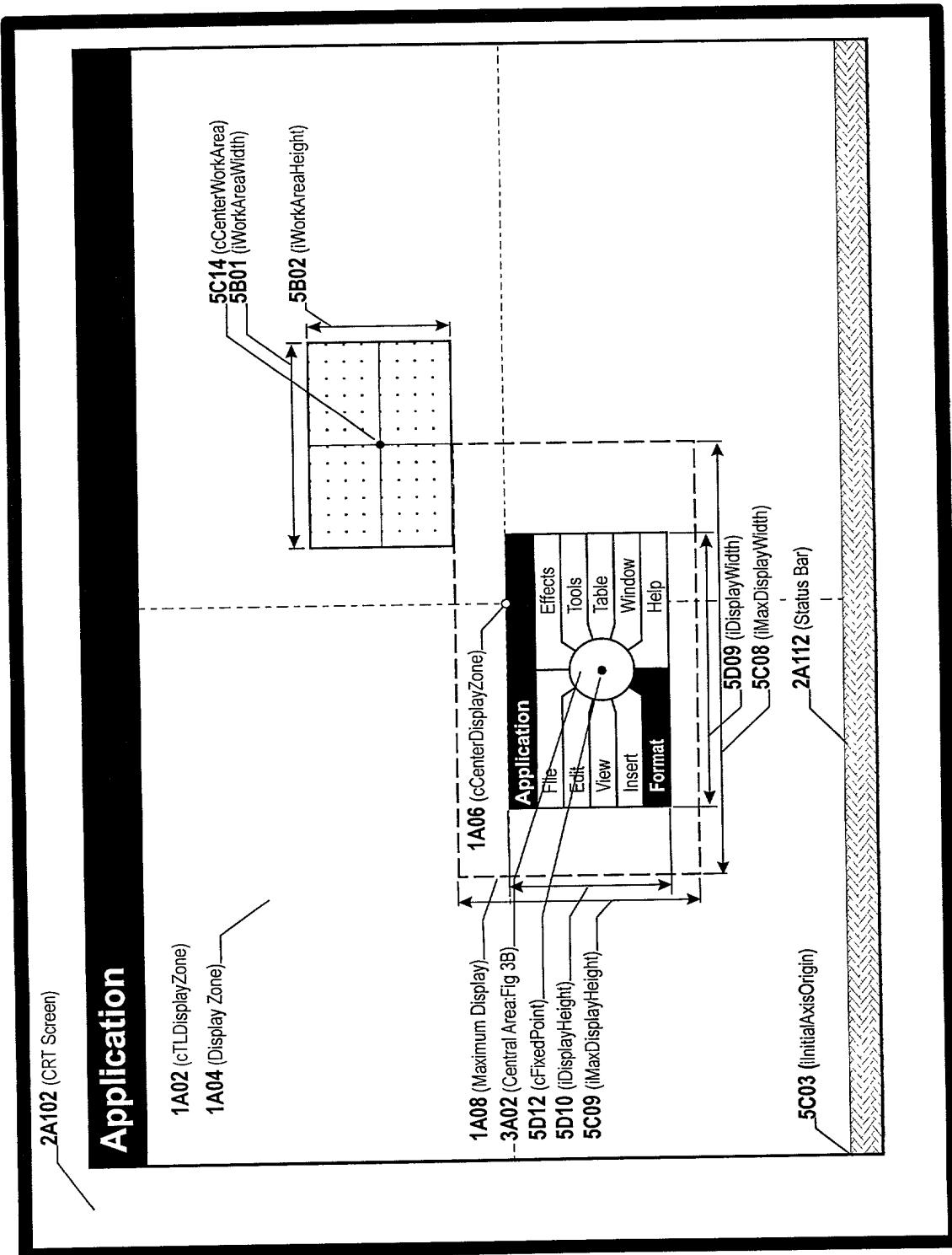


FIGURE 2A3

Application**FIGURE 2A4**

**FIGURE 3A1**

Application

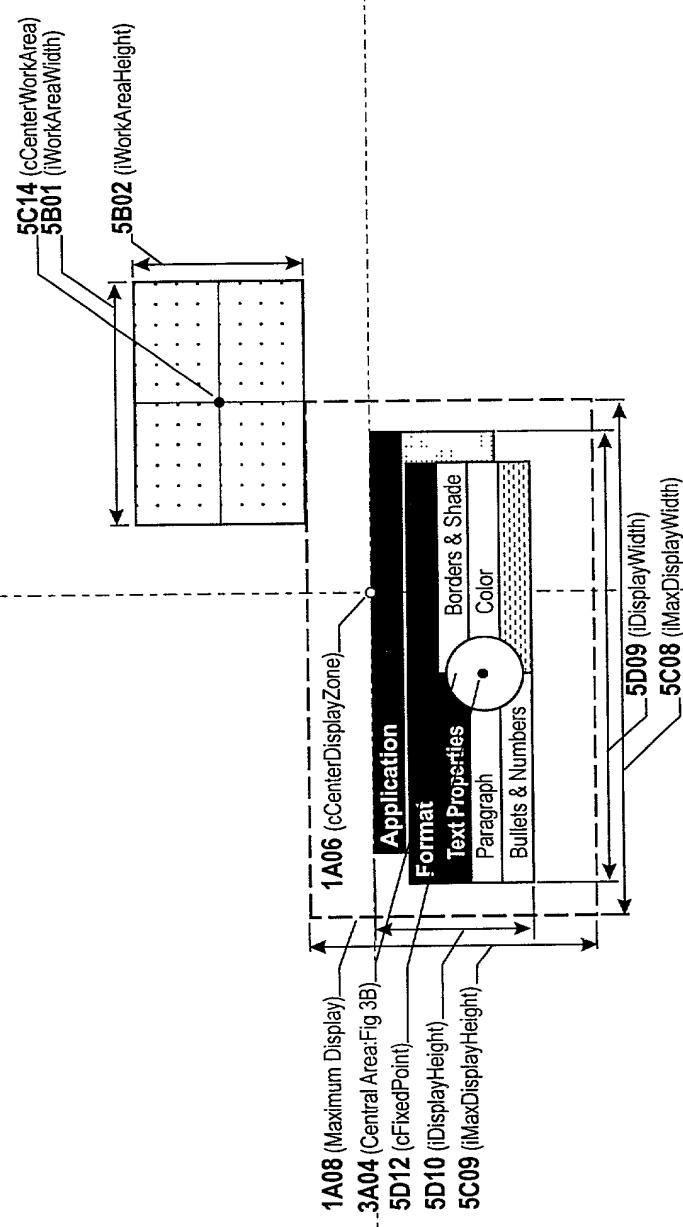


FIGURE 3A2

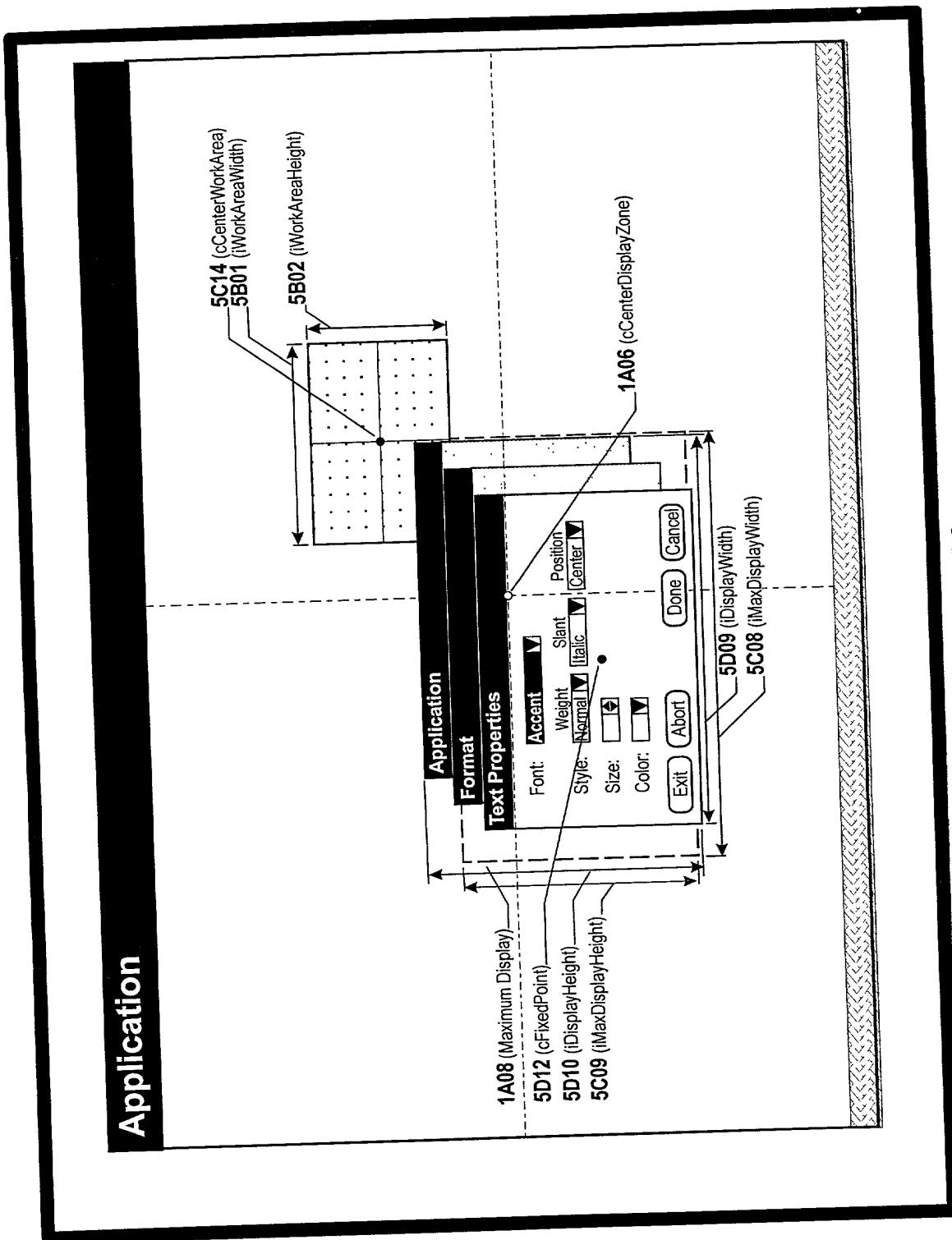


FIGURE 3A3

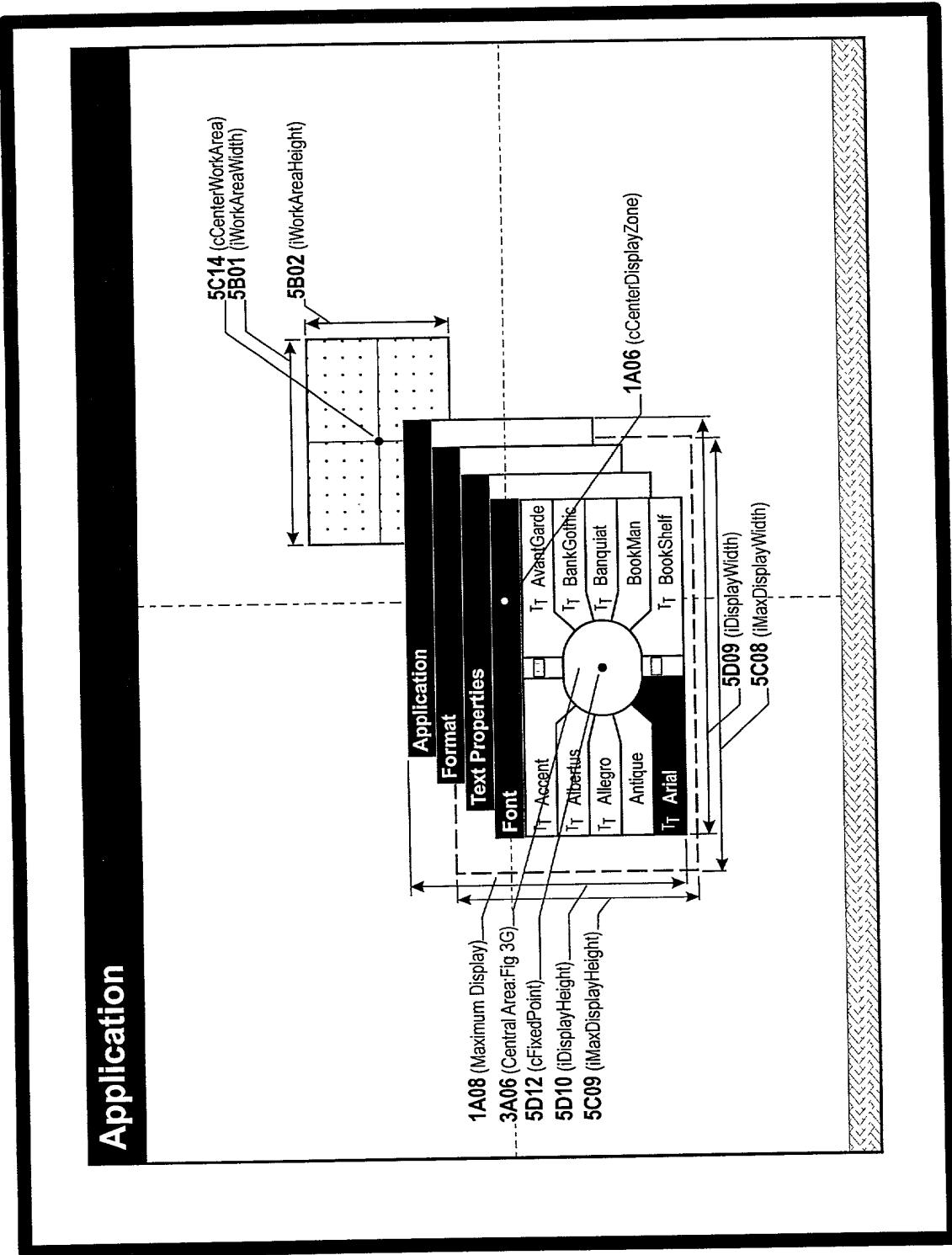
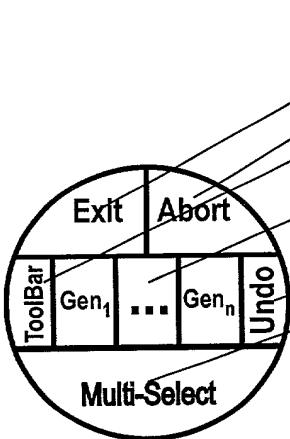
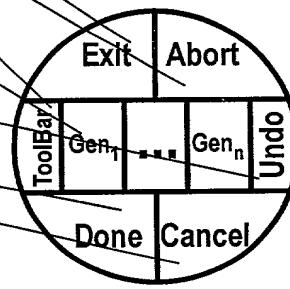
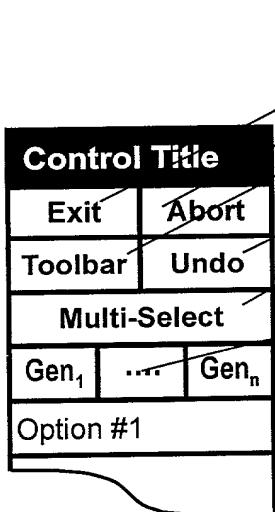
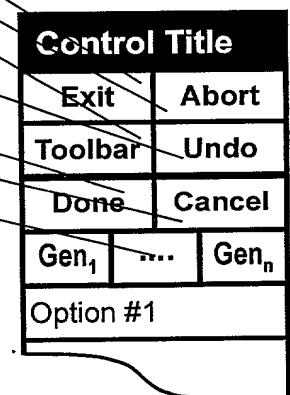
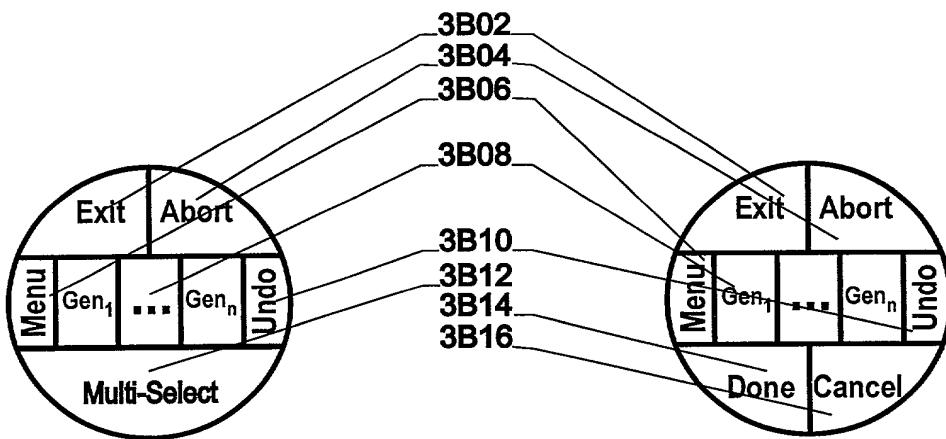
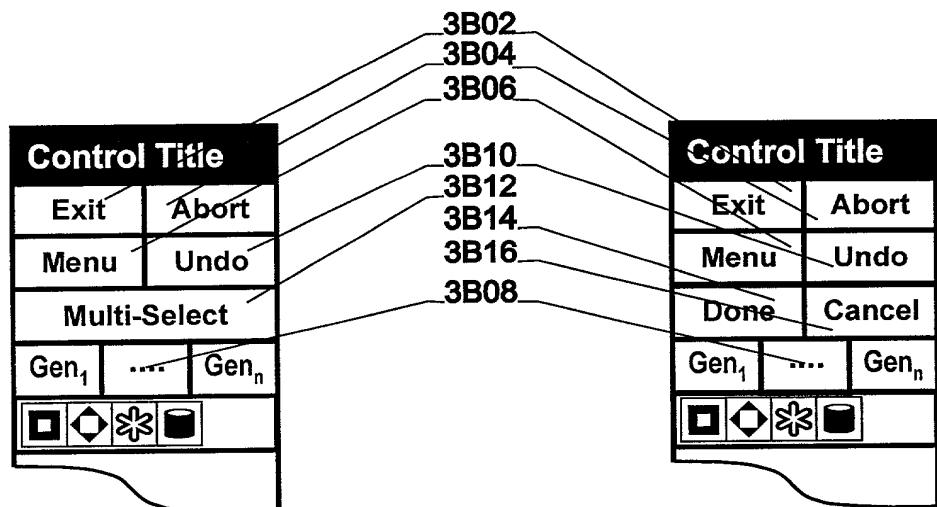
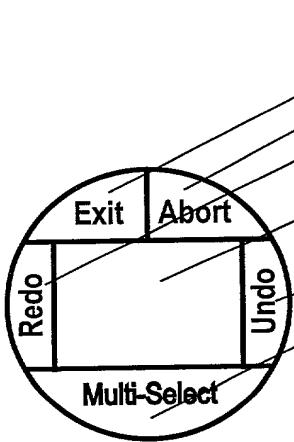
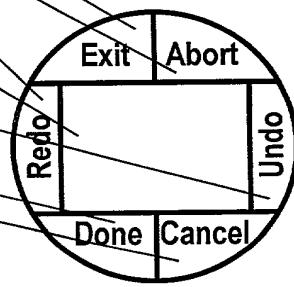
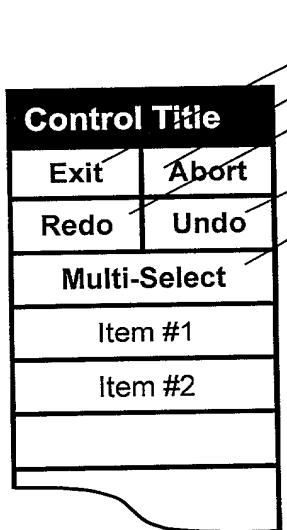
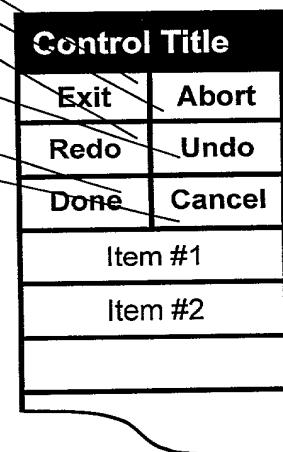


FIGURE 3A4

**FIGURE 3B1****FIGURE 3B2****FIGURE 3B3****FIGURE 3B4**

**FIGURE 3C1****FIGURE 3C2****FIGURE 3C3****FIGURE 3C4**

**FIGURE 3D1****FIGURE 3D2****FIGURE 3D3****FIGURE 3D4**

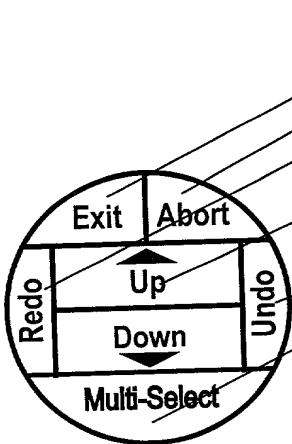


FIGURE 3E1

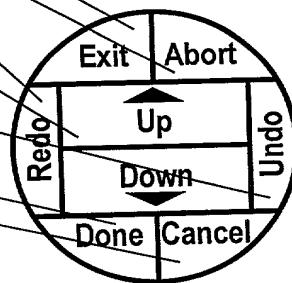


FIGURE 3E2

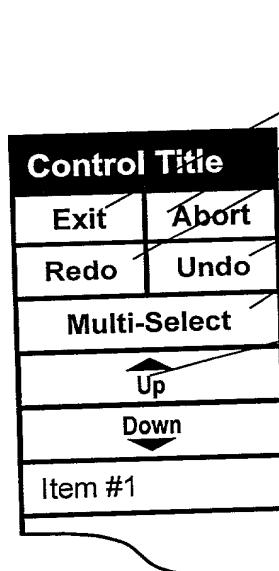


FIGURE 3E3

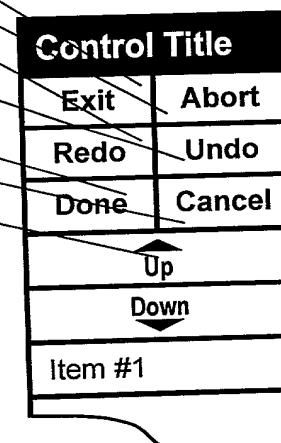
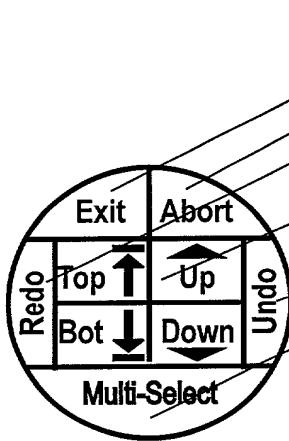
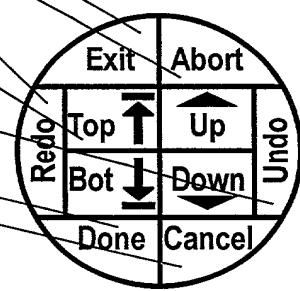
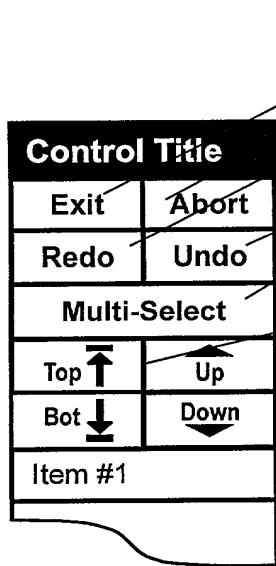
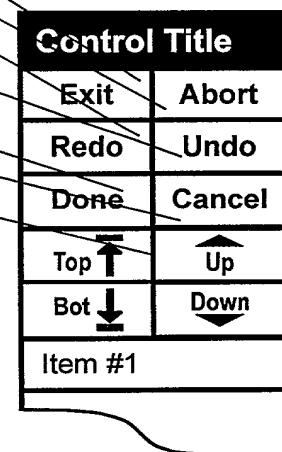


FIGURE 3E4

**FIGURE 3F1****FIGURE 3F2****FIGURE 3F3****FIGURE 3F4**

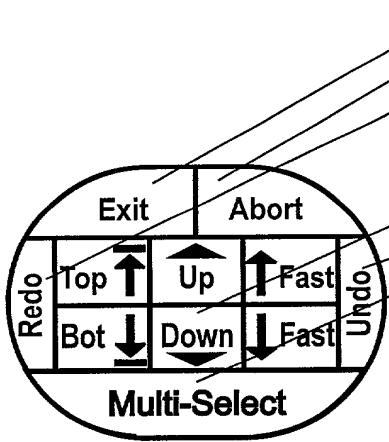


FIGURE 3G1

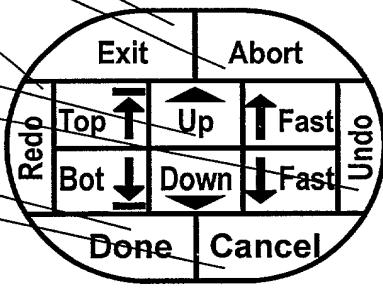


FIGURE 3G2



FIGURE 3G3

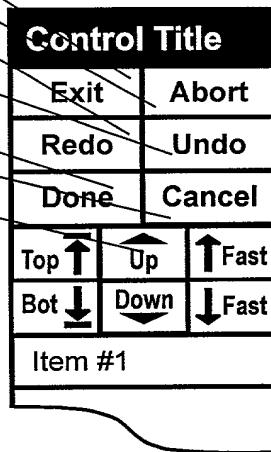


FIGURE 3G4

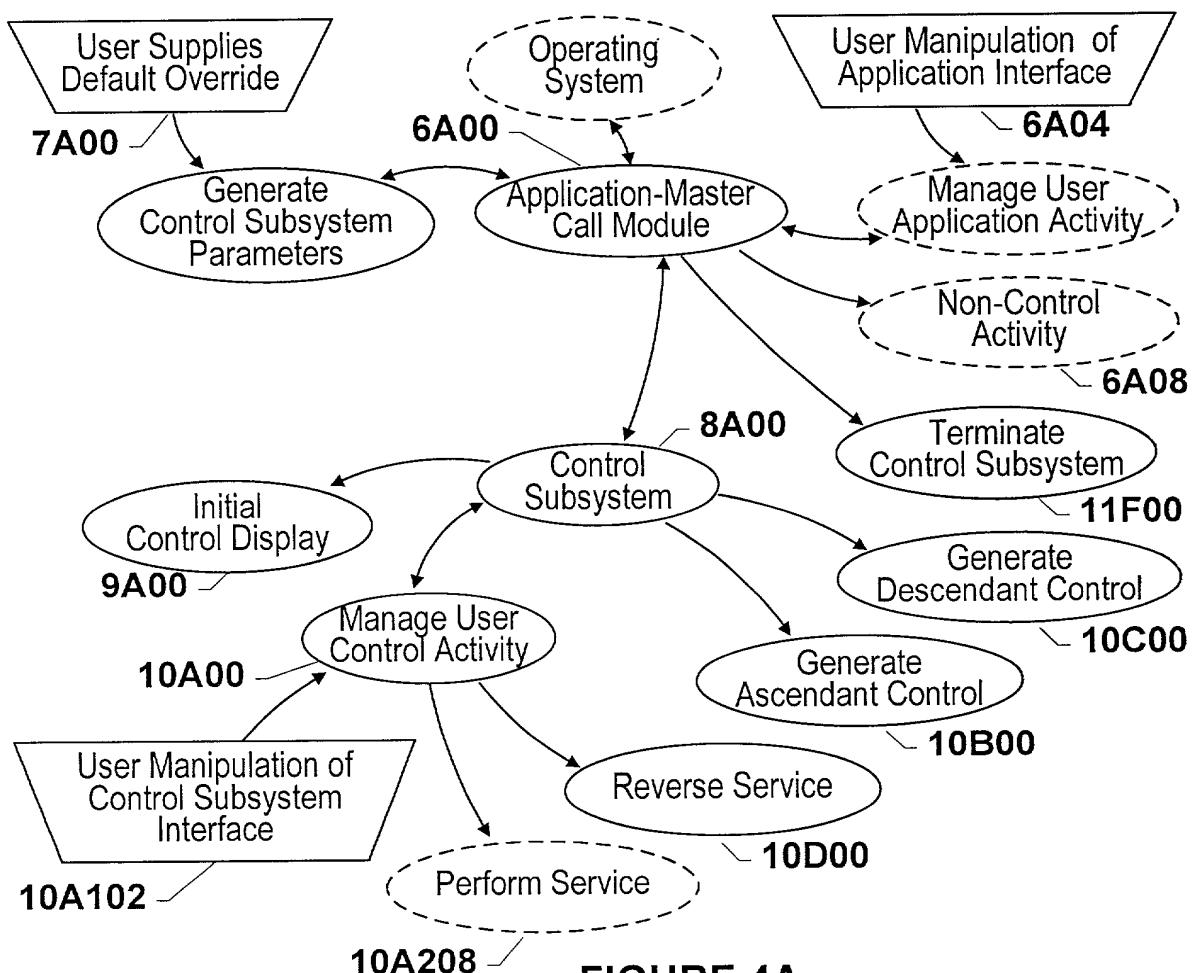


FIGURE 4A

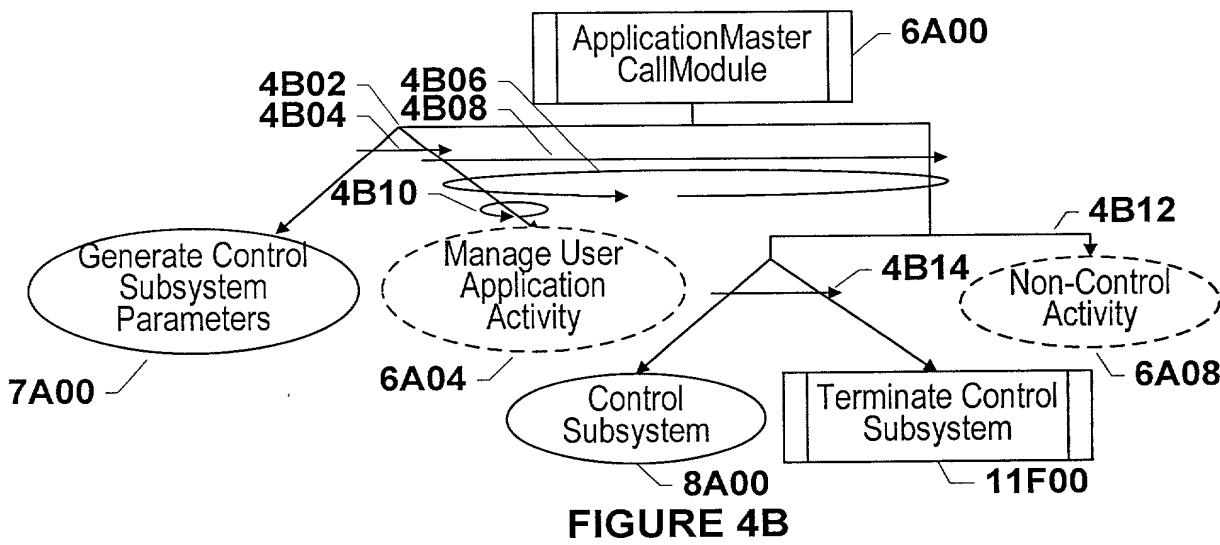
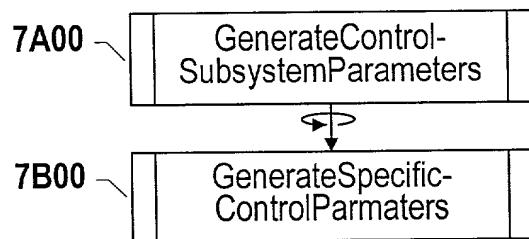
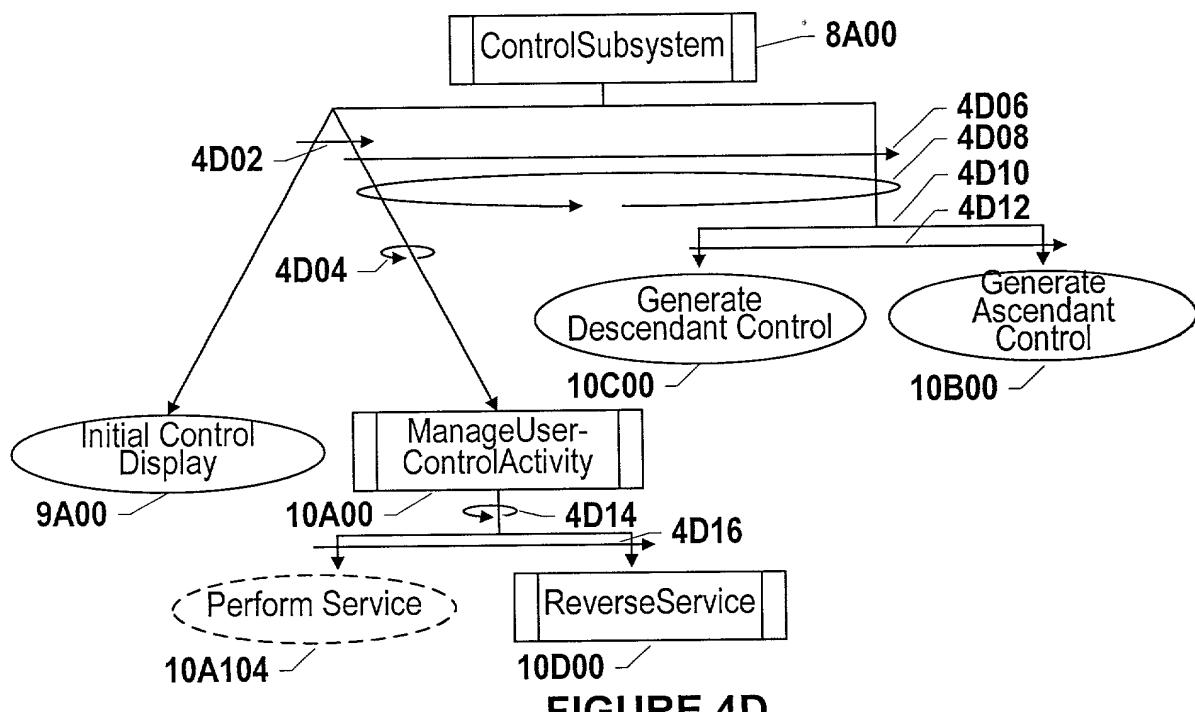


FIGURE 4B

**FIGURE 4C****FIGURE 4D**

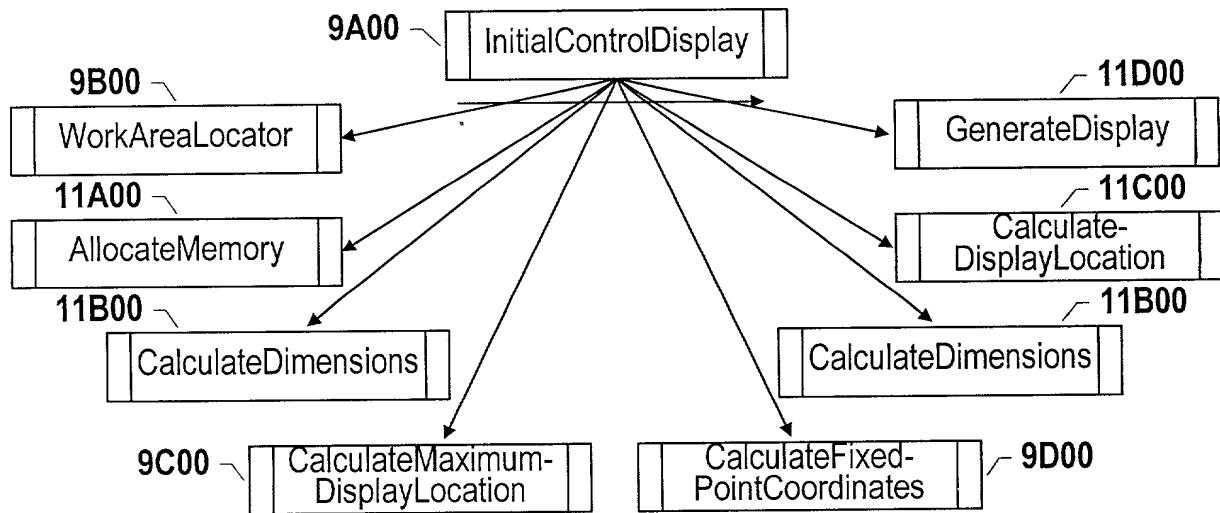


FIGURE 4E

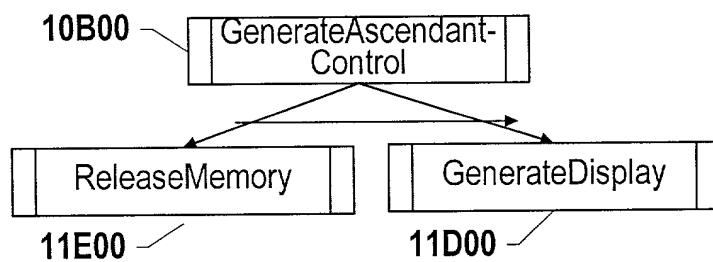


FIGURE 4F

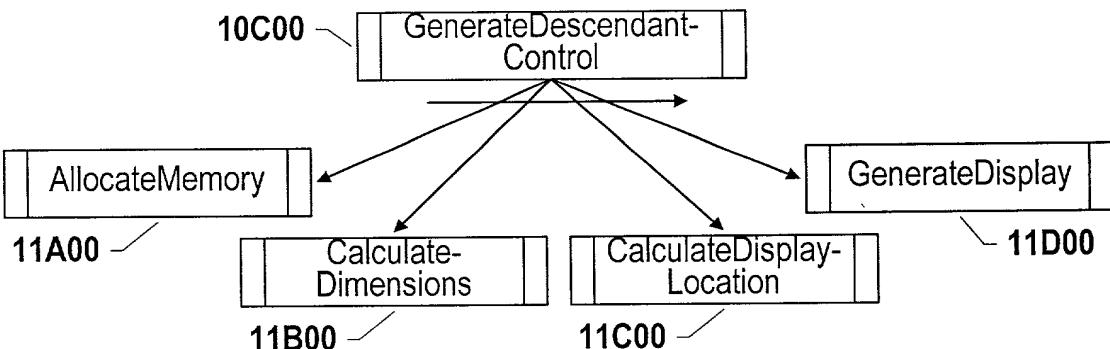


FIGURE 4G

rDefaultParms

5A00

rCommon	aControl
---------	----------

rCommon

01	iWorkAreaWidth
02	iWorkAreaHeight
03	rDisplayDisplace_X
04	rDisplayDisplace_Y
05	sFormat
06	iTitleBarHeight
07	iCascadeDisplace_X
08	iCascadeDisplace_Y
09	bCursorJump
10	rWAx/DZxRatio
11	rWAy/DZyRatio
12	bLikelyMaxDimensions
13	iMaxServices
14	(unused)

aControl[sControlType]

rVar[Menu]
rVar[ToolBar]
rVar[ListBox]
rVar[Dialog]
rVar[Other]

rVar

15	iRegionHeight
16	iRegionsLimit
17	iIconWidth
18	iMaxLabelLength
19	iMaxKeyEquivLength
20	iInterGap
21	iMaxNumGenerations
22	rFixedPointDisplace_X
23	rFixedPointDisplace_Y
24	iMaxShellWidth
25	iMaxShellHeight
26	iMaxDisplayWidth
27	iMaxDisplayHeight
28	pBaseControlParms
29	iBaseShellWidth
30	iBaseShellHeight
31	iBaseDisplayWidth
32	iBaseDisplayHeight

DEFINITIONS: See definitions of Figure 5B

FIGURE 5A

rDisplayParms		5B00
	rCommon	aControl
rCommon		aControl[sControlType]
01	iWorkAreaWidth	rVar[Menu]
02	iWorkAreaHeight	rVar[ToolBar]
03	rDisplayDisplace_X	rVar[ListBox]
04	rDisplayDisplace_Y	rVar[Dialog]
05	sFormat	rVar[Other]
06	iTitleBarHeight	
07	iCascadeDisplace_X	
08	iCascadeDisplace_Y	
09	bCursorJump	
10	rWAx/DZxRatio	
11	rWAy/DZyRatio	
12	bLikelyMaxDimensions	
13	iMaxServices	
14	(unused)	
		rVar
15	iRegionHeight	
16	iRegionsLimit	
17	iIconWidth	
18	iMaxLabelLength	
19	iMaxKeyEquivLength	
20	iInterGap	
21	iMaxNumGenerations	
22	rFixedPointDisplace_X	
23	rFixedPointDisplace_Y	
24	iMaxShellWidth	
25	iMaxShellHeight	
26	iMaxDisplayWidth	
27	iMaxDisplayHeight	
28	pBaseControlParms	
29	iBaseShellWidth	
30	iBaseShellHeight	
31	iBaseDisplayWidth	
32	iBaseDisplayHeight	

FIGURE 5B1

DEFINITIONS

rCommon: PARAMETER VALUES APPLICABLE TO ALL CONTROLS

01 iWorkAreaWidth	Width of implicit WorkArea.
02 iWorkAreaHeight	Height of implicit Work Area.
03 rDisplayDisplace_X	The per cent of one-half the work area width from which the reference corner of the maximim display is displaced from the Work Area center: -- Positive displacement is toward the Display Zone center. -- Negative displacement is away from the Display Zone center.
04 rDisplayDisplace_Y	The per cent of one-half the work area height from which the reference corner of the maximim display is displaced from the Work Area center. -- Positive displacement is toward the Display Zone center. -- Negative displacement is away from the Display Zone center.
05 sFormat	String identifying display format: "Traditional", "Spider", "Dialog" or "Other".
06 iTitleBarHeight	Height of title bar.
07 iCascadeDisplace_X	Horizontal displacement with which each successive ancestor of the control display is cascaded.
08 iCascadeDisplace_Y	Vertical displacement with which each successive of ancestorof the control display is cascaded.
09 bJumpCursor	Default disposal of cursor at termination of control display: -- TRUE -> DO move cursor to cStartCurtosr coordinates. -- FALSE-> DO NOT move cursor to cStartCurtosr coordinates.
10 rWAx/DZxRatio	Ratio of work-area/display-zone width that sets rDisplayDisplace_Y=0.0 and rDisplayDisplace_X=1.0;
11 rWAz/DZzRatio	Ratio of work-area/display-zone height that sets rDisplayDisplace_Y=1.0 and rDisplayDisplace_X=0.0;
12 bLikelyMaxDimensions	TRUE -> User supplies dimensions of likely largest control display, FALSE -> User supplies dimensions of the absolutely maximum display.
13 iMaxServices	Maximum number of services that will ever ever requested during a Control Subsystem activation.

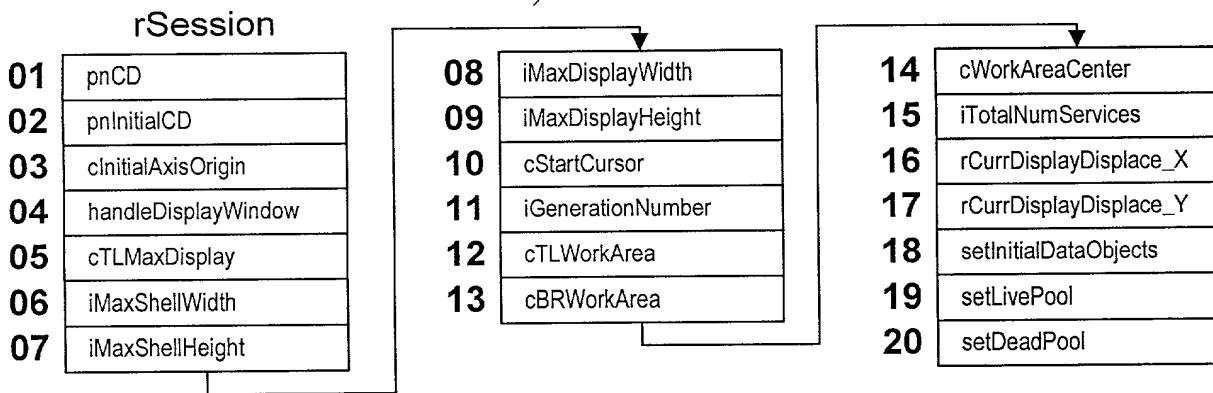
FIGURE 5B2

DEFINITIONS (Continued)

aControl[sFormat].rVar:	PARAMETER VALUES SPECIFIC TO EACH CONTROL MANAGED
15 iRegionHeight	Height of area of display in which an item or a group of items is displayed.
16 iRegionsLimit	Max number of different items or groups of related items appearing on any single display.
17 ilconWidth	Width of any icon present.
18 iMaxLabelLength	Maximum length of any label appearing in a control.
19 iMaxKeyEquivLength	Maximum length of any key-equivalent symbols appearing in a control.
20 iInterGap	Spacing between identification elements in a region
21 iMaxNumGenerations	Maximum number of controls that can be present in a control path.
22 rFixedPointDisplace_X	Percent of iMaxShellWidth the cFixedFocusPoint.X is displaced rightward from top-left corner of the maximum shell.
23 rFixedPointDisplace_Y	Percent of iMaxShellHeight the cFixedFocusPoint.Y is displaced downward from top-left corner of the maximum shell.
24 iMaxShellWidth	Maximum width required for Menu & ToolBar shell. Null for other controls
25 iMaxShellHeight	Maximum Height required for Menu & ToolBar shell. Null for other controls.
26 iMaxDisplayWidth	Maximum width required for Menu & ToolBar display. Null for other controls
27 iMaxDisplayHeight	Maximum Height required for Menu & ToolBar display. Null for other controls.
28 pBaseControlParms	Pointer to the data structure defining the initial menu or toolbar displayed at control activation. This parameter is null for controls having parameters that can vary during different activations.
29 iBaseShellWidth	Width of initial Menu & ToolBar shell. Null for other controls
30 iBaseShellHeight	Height of initial Menu & ToolBar shell. Null for other controls.
31 iBaseDisplayWidth	Width of initial Menu & ToolBar display. Null for other controls
32 iBaseDisplayHeight	Height of initial Menu & ToolBar display. Null for other controls.

FIGURE 5B3

5C00

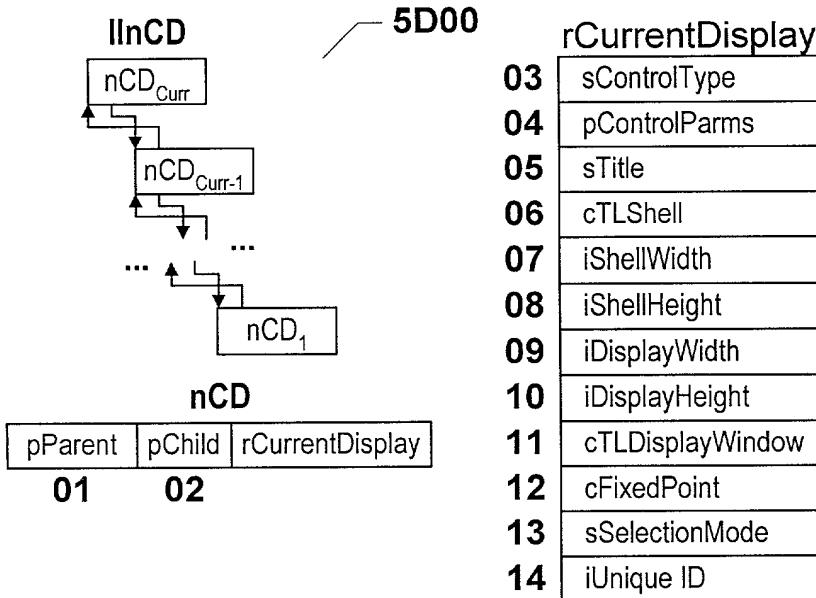


DEFINITIONS

rCurrentSession: VARIABLES OF CURRENT ACTIVATION OF DISPLAY SUB-SYSTEM

- 01** pnCD Pointer to node of llnCD storing values fo the current display.
- 02** plnitialnAC Pointer to first node of the llnCD data structure.
- 03** cInitialAxisOrigin Screen coordinate pair of defining the axis origin at control subsystem activation.
- 04** handleDisplayWindow Pointer to data and procedures controlled by the operating system that manage the window containing the controldisplay.
- 05** cTLMaxDisplay Coordinate pair defining best location of top-left corner of the maximum display.
- 06** iMaxShellWidth Width of max shell that can be encountered given the initial display requested.
- 07** iMaxShellHeight Height of max shell that can be encountered given the initial display requested.
- 08** iMaxDisplayWidth Width of max display that can be encountered given the initial display requested.
- 09** iMaxDisplayHeight Height of max display that can be encountered given the initial display requested.
- 10** cStartCursor Coordinate pair defining the cursor location relative to cInitialAxisOrigin at control subsystem activation.
- 11** iGenerationNumber Number of generations of current toolbar display.
- 12** cTLWorkArea Coordinate pair defining the top-left corner of the current work area.
- 13** cBRWorkArea Coordinates of bottom-right corner of the current work area.
- 14** cWorkAreaCenter Coordinate pair defining the center of the current work area.
- 15** iTotalNumServices Number of services performed during the current control subsystem activation.
- 16** rCurrDisplayDisplace_X rDisplayDisplace_X parameter with allowance for extreme work area dimensions.
- 17** rCurrDisplayDisplace_Y rDisplayDisplace_Y parameter with allowance for extreme work area dimensions.
- 18** setInitialDataObjects Set of data objects existing control subsystem activated.
- 19** setLivePool Accumulating set of data objects created by user manipulation of controls during current activation of the control subsystem.
- 20** setDeadPool Accumulating set of data objects destroyed by user manipulation of controls during current activation of the control subsystem.

FIGURE 5C



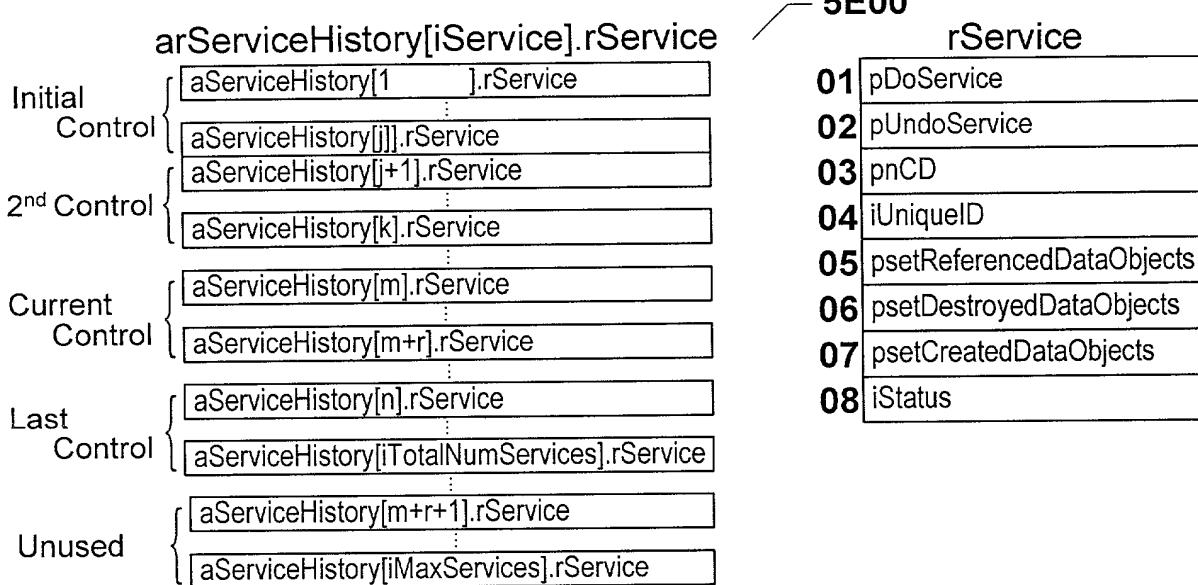
DEFINITIONS

- 01** pParent Pointer to parent of current nCD node
02 pChild Pointer to child of current nCD node.

rCurrentDisplay: VARIABLE VALUES TO MANAGE CURRENT DISPLAY

- 03** sControlType Type of control managed by current lInCD node:
 "Menu", "ToolBar", "ListBox", "Dialog", "Other".
- 04** pControlParms Pointer to system data structure defining parameters of the control identified by the current nCD node.
- 05** sTitle String containing label appearing inTitleBar
- 06** cTLShell Coordinate pair defining the top-left corner of the shell of the current control.
- 07** iShellWidth Width of the current shell.
- 08** iShellHeight Height of the current shell.
- 09** iDisplayWidth Width of the current display.
- 10** iDisplayHeight Height of the current display.
- 11** cTLDisplayWindow Coordinate pair defining the top-left corner of the current display.
- 12** cFixedPoint Coordinate pair defining the current fixed point.
- 13** sSelectionMode Designates kind of selection from control is permitted
 - Single -> single selection is imposed by design.
 - User ->"MultiSelect" displayed.
 - * If "Multi-Sect" not selected, user is constrained to single selection.
 - * If "Multi-Sect" selected, the multi-select area is reconfigured to "Done|Cancel", and the user is not constrained to single selection..
 - Multiple -> multiple selection is permitted.
- 14** iUniqueId The unique identification number of specific control defined by current lInCD node.

FIGURE 5D



DEFINITIONS

rService: VARIABLE VALUES TO MANAGE ARBITRARY UNDO

- 01 pDoService** Pointer to processes that perform the Xth service of the current Control Subsystem activation.
- 02 pUndoService** Pointer to processes that perform the reverse capability of the Xth service of the current Control Subsystem activation.
- 03 pnCD** Pointer to llnCD node defining the control from which the Xth service of the current control subsystem activation was requested.
- 04 iUniqueID** Unique identification number of the control identified by the current node.
- 05 psetReferencedDataObjects** Pointer to a set identifying all data objects utilized during performance of the Xth service excluding reference to any data object(s) created by that service.
- 06 psetDestroyedDataObjects** Pointer to a set identifying all data objects destroyed during performance of the Xth service.
- 07 psetCreatedDataObjects** Pointer to a set identifying all data objects created during performance of the Xth service.
- 08 iStatus** Indicator to the current state of the Xth service:
 - +1 -> service has not been reversed
 - 0 -> service is temporarily reversed
 - 1 -> service currently reversed because of dependence on a one or more data objects created by prior service(s) that have been reversed.
 - 2 -> service is permanently reversed

FIGURE 5E

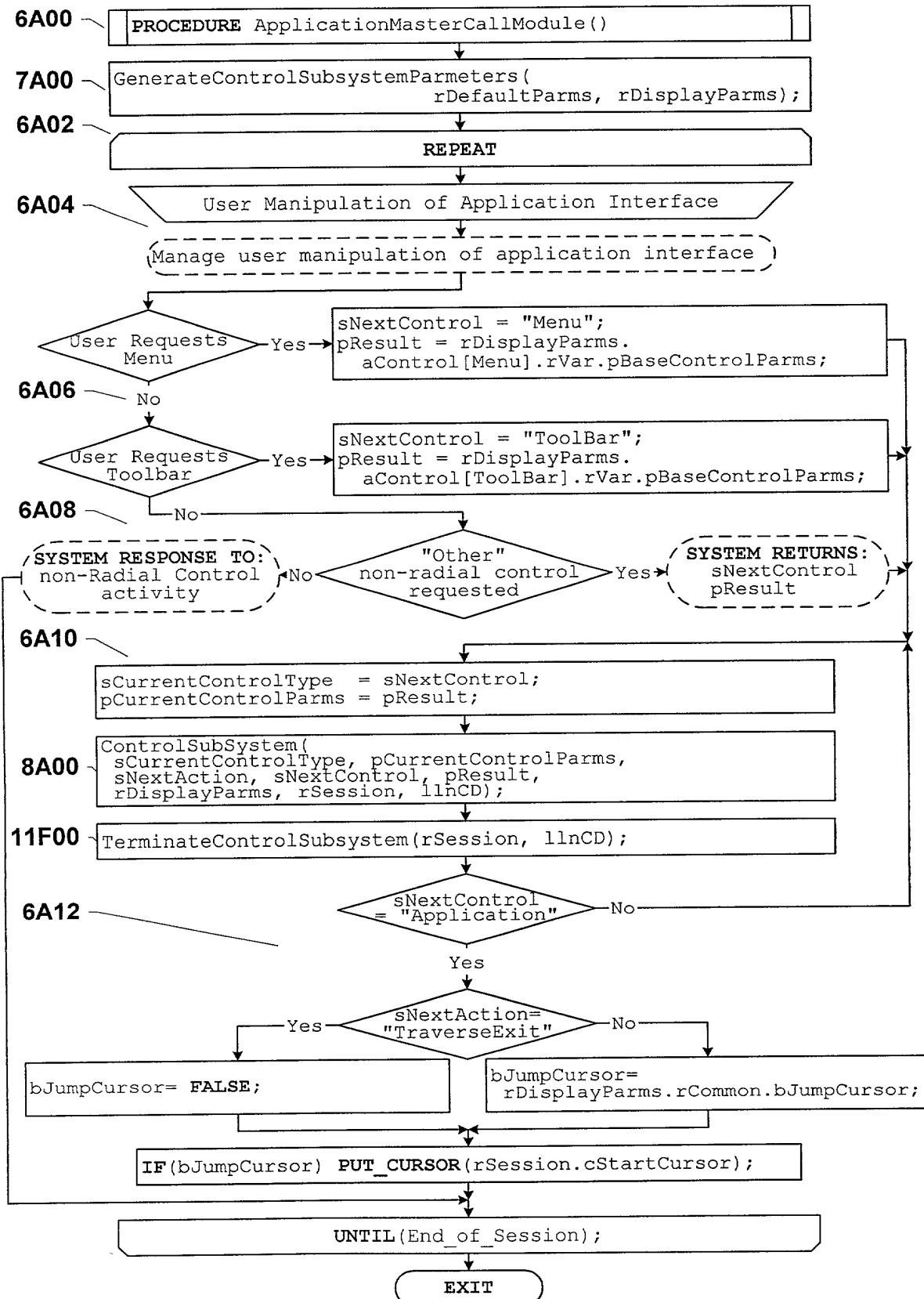


FIGURE 6A



FIGURE 7A1

7B00

```
PROCEDURE GenerateSpecificControlParameters(
    sControlType, rDefaultParms, rDisplayParms);
```

7B02

```
WITH rDisplayParms.aControl[sControlType].rVar
// Designers submit specification of control graphic (null if NA)
    iRegionHeight =
        rDefaultParms.aControl[sControlType].rVar.iRegionHeight;
    iRegionsLimit =
        rDefaultParms.aControl[sControlType].rVar.iRegionsLimit;
    iIconWidth =
        rDefaultParms.aControl[sControlType].rVar.iIconWidth ;
    iMaxLabelLength =
        rDefaultParms.aControl[sControlType].rVar.iMaxLabelLength;
    iMaxKeyEquivLength= i
        rDefaultParms.aControl[sControlType].rVar.iMaxKeyEquivLength;
    iInterGap =
        rDefaultParms.aControl[sControlType].rVar.iInterGap;
```

7B04

```
// Designers submit for maximum number of generations
    iMaxNumGenerations =
        rDefaultParms.aControl[sControlType].rVar.iMaxNumGenerations;
```

7B06

```
// Designers submit specification for fixed-point displacement
    iFixedPointDisplace_X =
        rDefaultParms.aControl[sControlType].rVar.iFixedPointDisplace_X;
    iFixedPointDisplace_Y =
        rDefaultParms.aControl[sControlType].rVar.iFixedPointDisplace_Y;
```

7B08

```
// Designers submit specification for maximum dimensions
    iMaxShellWidth =
        rDefaultParms.aControl[].rVar.iMaxShellWidth;
    iMaxShellHeight=
        rDefaultParms.aControl[sControlType].rVar.iMaxShellHeight;
    iMaxDisplayWidth =
        rDefaultParms.aControl[sControlType].rVar.iMaxDisplayWidth;
    iMaxDisplayHeight=
        rDefaultParms.aControl[sControlType].rVar.iMaxDisplayHeight;
```

7B10

```
// Designers submit specification for basic dimensions
    pBaseControlParms=
        rDefaultParms.aControl[sControlType].rVar.pBaseControlParms;
    iBaseShellWidth =
        rDefaultParms.aControl[sControlType].rVar.iBaseShellWidth;
    iBaseShellHeight=
        rDefaultParms.aControl[sControlType].rVar.iBaseShellHeight;
    iBaseDisplayWidth =
        rDefaultParms.aControl[sControlType].rVar.iBaseDisplayWidth;
    iBaseDisplayHeight=
        rDefaultParms.aControl[sControlType].rVar.iBaseDisplayHeight;
END_WITH
```

RETURN

FIGURE 7B1

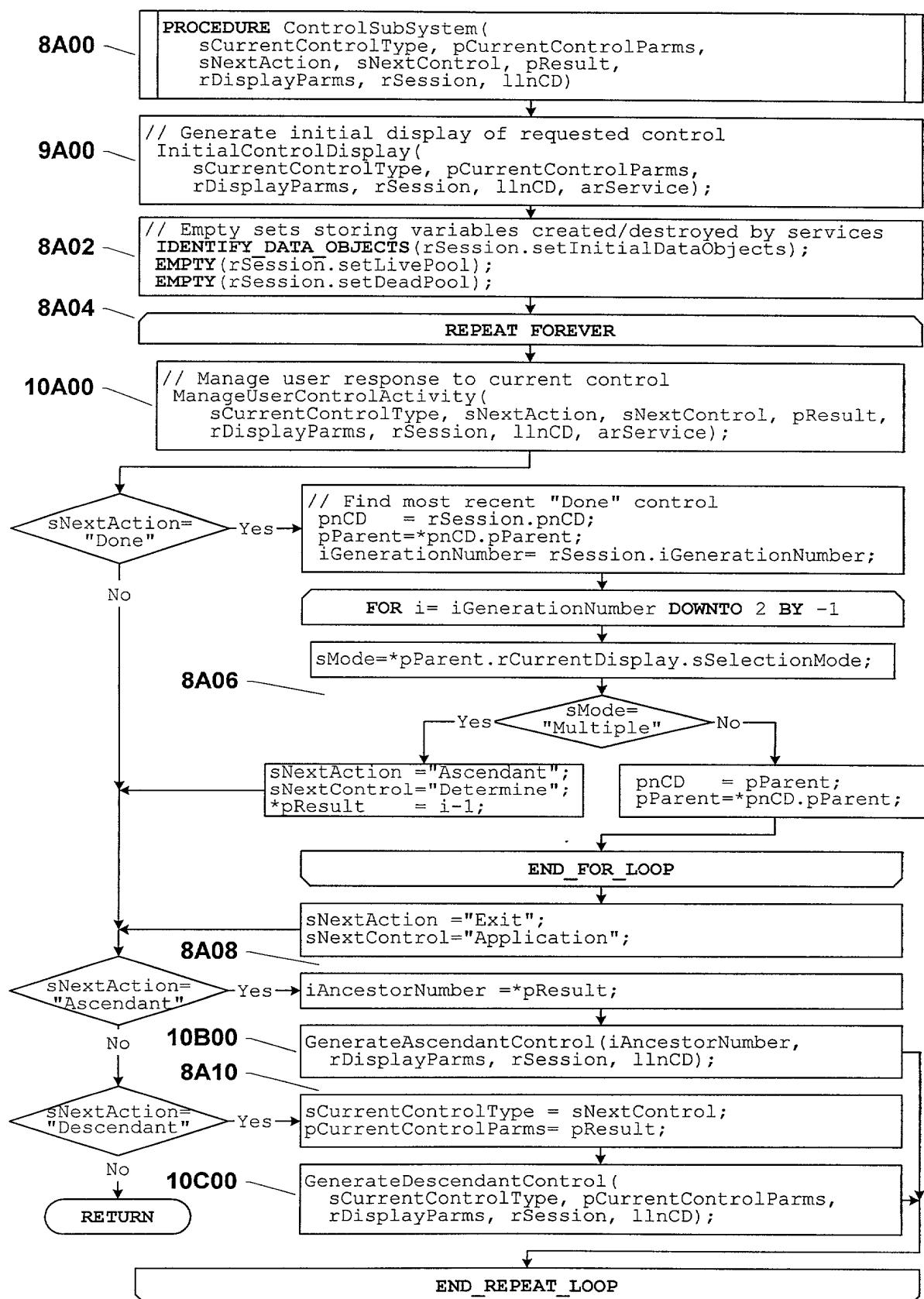


FIGURE 8A

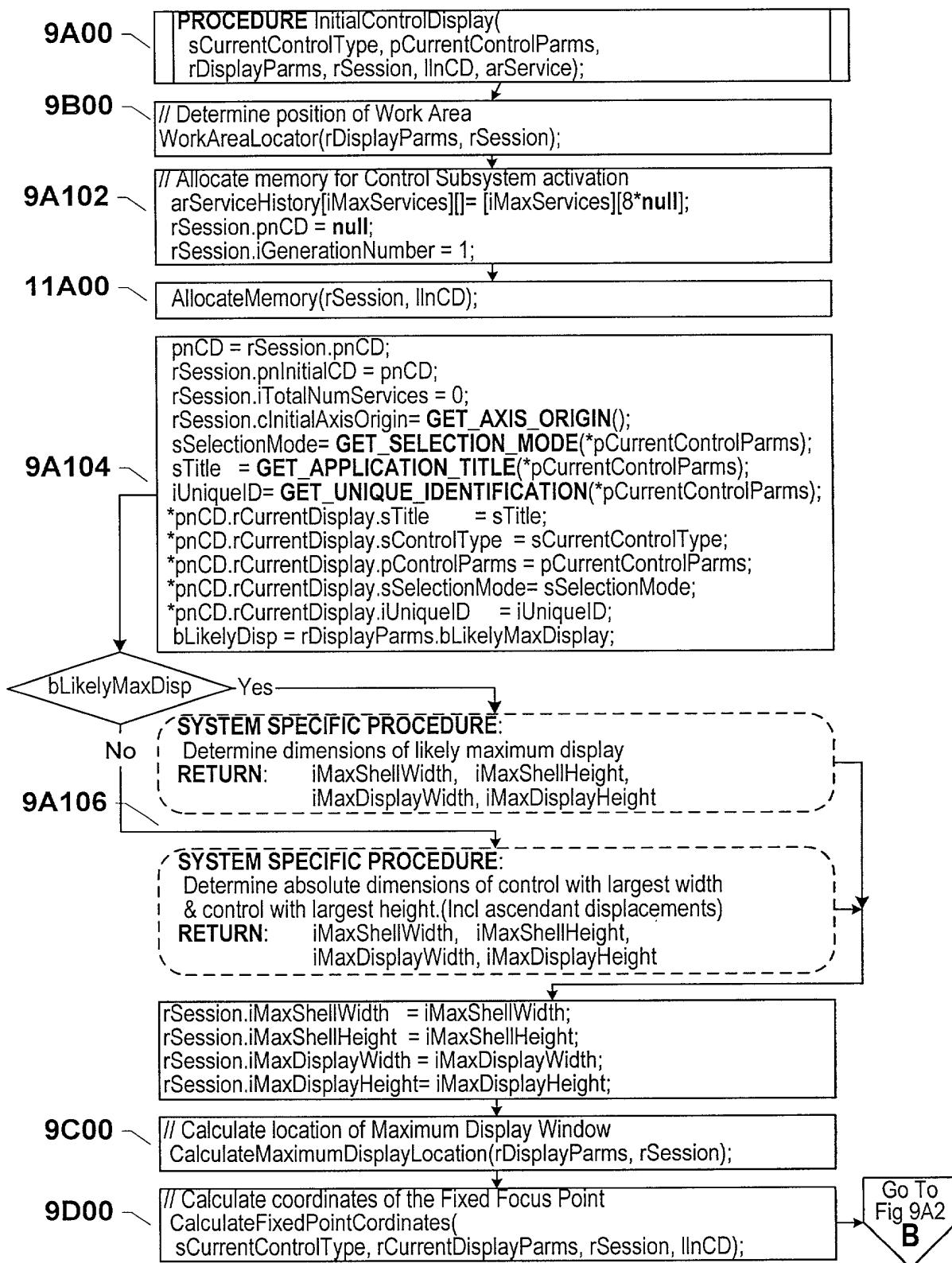


FIGURE 9A1

33/61

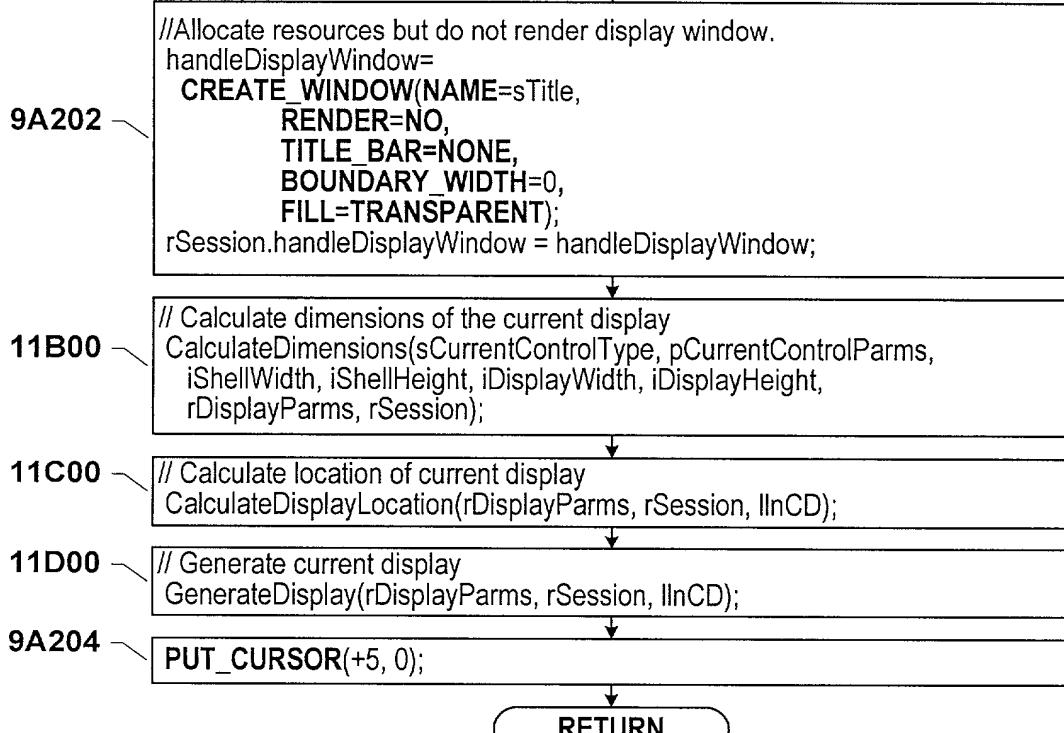


FIGURE 9A2

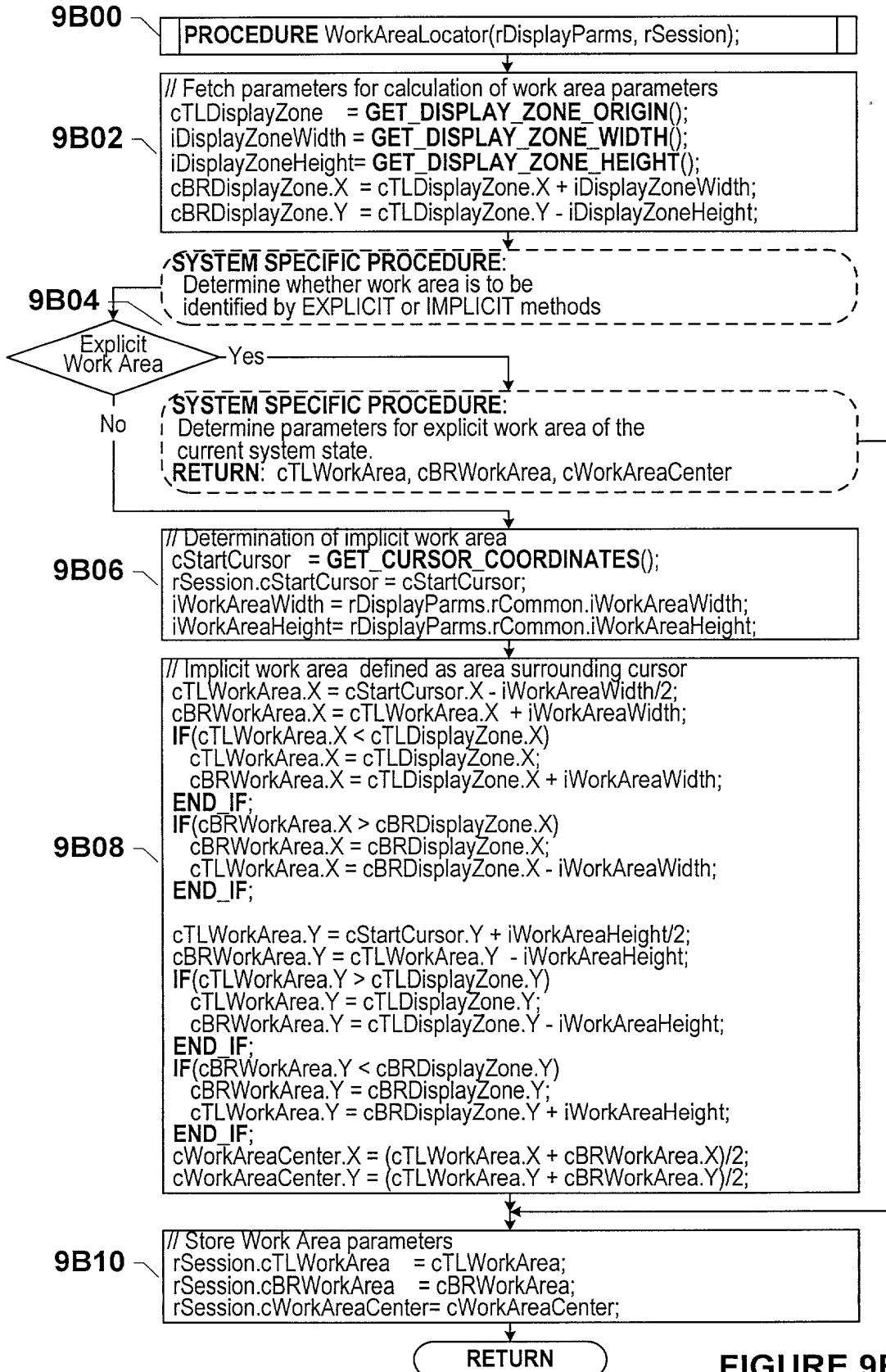


FIGURE 9B

PROGRAM CalculateMaximumDisplayLocation(rDisplayParms, rSession);

9C00

```
cTLDisplayZone = GET_DISPLAY_ZONE_ORIGIN();
iDisplayZoneWidth = GET_DISPLAY_ZONE_WIDTH();
iDisplayZoneHeight = GET_DISPLAY_ZONE_HEIGHT();
cBRDisplayZone.X = cTLDisplayZone.X + iDisplayZoneWidth;
cBRDisplayZone.Y = cTLDisplayZone.Y - iDisplayZoneHeight;
```

9C102

```
//Determine parameters to position Max Display reference corner
iMaxDisplayWidth = rSession.cMaxDisplayWidth;
iMaxDisplayHeight = rSession.cMaxDisplayHeight;
cTLWorkArea = rSession.cTLWorkArea;
cBRWorkArea = rSession.cBRWorkArea;
iWorkAreaWidth = cBRWorkArea.X - cTLWorkArea.X;
iWorkAreaHeight = cTLWorkArea.Y - cBRWorkArea.Y;
rWAX/DZxRatio = rDisplayParms.rCommon.rWAX/DZxRatio;
rWAY/DZyRatio = rDisplayParms.rCommon.rWAY/DZyRatio;
IF(iWorkAreaWidth/iDisplayZoneWidth > rWAX/DZxRatio)
    rDisplayDisplace_X= 0.0;    rDisplayDisplace_Y= 1.0;
ELSEIF(iWorkAreaHeight/iDisplayZoneHeight > rWAY/DZyRatio)
    rDisplayDisplace_X= 1.0;    rDisplayDisplace_Y= 0.0;
ELSE
    rDisplayDisplace_X= rDisplayParms.rCommon.rDisplayDisplace_X;
    rDisplayDisplace_Y= rDisplayParms.rCommon.rDisplayDisplace_Y;
END_IF;
rSession.rCurrDisplayDisplace_X= rDisplayDisplace_X;
rSession.rCurrDisplayDisplace_Y= rDisplayDisplace_Y;
```

9C104

```
// Determine direction of Center Seeking:
IF((cWorkAreaCenter.X >= cDisplayZoneCenter.X) AND
(cWorkAreaCenter.Y >= cDisplayZoneCenter.Y))
    sCenterSeeking= "DownLeft";
IF((cWorkAreaCenter.X >= cDisplayZoneCenter.X) AND
(cWorkAreaCenter.Y < cDisplayZoneCenter.Y))
    sCenterSeeking= "UpLeft";
IF((cWorkAreaCenter.X < cDisplayZoneCenter.X) AND
(cWorkAreaCenter.Y >= cDisplayZoneCenter.Y))
    sCenterSeeking= "DownRight";
IF((cWorkAreaCenter.X < cDisplayZoneCenter.X) AND
(cWorkAreaCenter.Y < cDisplayZoneCenter.Y));
    sCenterSeeking= "UpRight";
```

9C106

```
// Calc optimum location of max control display reference corner
iDisplace_X= rDisplayDisplace_X*WorkAreaWidth/2;
iDisplace_Y= rDisplayDisplace_Y*WorkAreaHeight/2;
CASE(sCenterSeeking)
    "DownLeft":   cOpt.X= cWorkAreaCenter.X-iDisplace_X;
                  cOpt.Y= cWorkAreaCenter.Y-iDisplace_Y;
    "UpLeft":     cOpt.X= cWorkAreaCenter.X-iDisplace_X;
                  cOpt.Y= cWorkAreaCenter.Y+iDisplace_Y;
    "DownRight":  cOpt.X= cWorkAreaCenter.X+iDisplace_X;
                  cOpt.Y= cWorkAreaCenter.Y-iDisplace_Y;
    "UpRight":    cOpt.X= cWorkAreaCenter.X+iDisplace_X;
                  cOpt.Y= cWorkAreaCenter.Y+iDisplace_Y;
END CASE:
```

9C108



FIGURE 9C1

```
// Calculate Free Space for Maximum Display
CASE(sCenterSeeking)
    "DownLeft": iFreeSpace.X = cOpt.X - cTLDisplayZone.X;
    iFreeSpace.Y = cOpt.Y - cBRDisplayZone.Y;
    "UpLeft": iFreeSpace.X = cOpt.X - cTLDisplayZone.X;
    iFreeSpace.Y = cTLDisplayZone.Y - cOpt.Y;
    "DownRight": iFreeSpace.X = cBRDisplayZone.X - cOpt.X;
    iFreeSpace.Y = cTLDisplayZone.Y - cOpt.Y;
    "UpRight": iFreeSpace.X = cBRDisplayZone.X - cOpt.X;
    iFreeSpace.Y = cOpt.Y - cBRDisplayZone.Y;
END_CASE;
```

A**9C202**

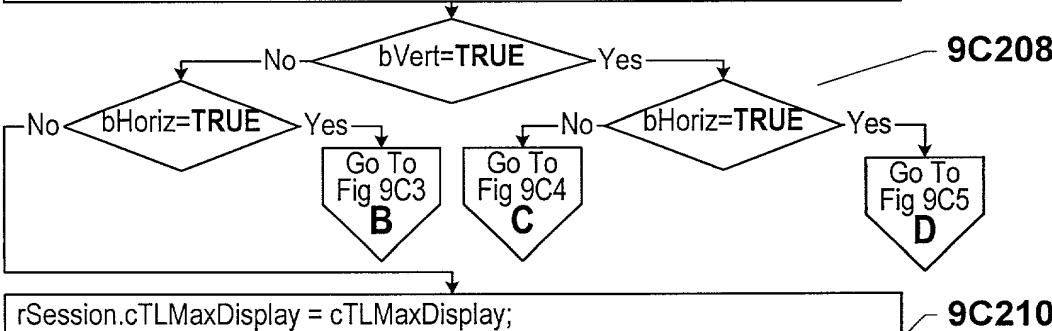
```
// Determine presence of constraint conditions:
IF(iFreeSpace_X < iMaxDisplayWidth) // Horizontal constraint?
    bHoriz = TRUE;
ELSE
    bHoriz = FALSE;
END_IF;

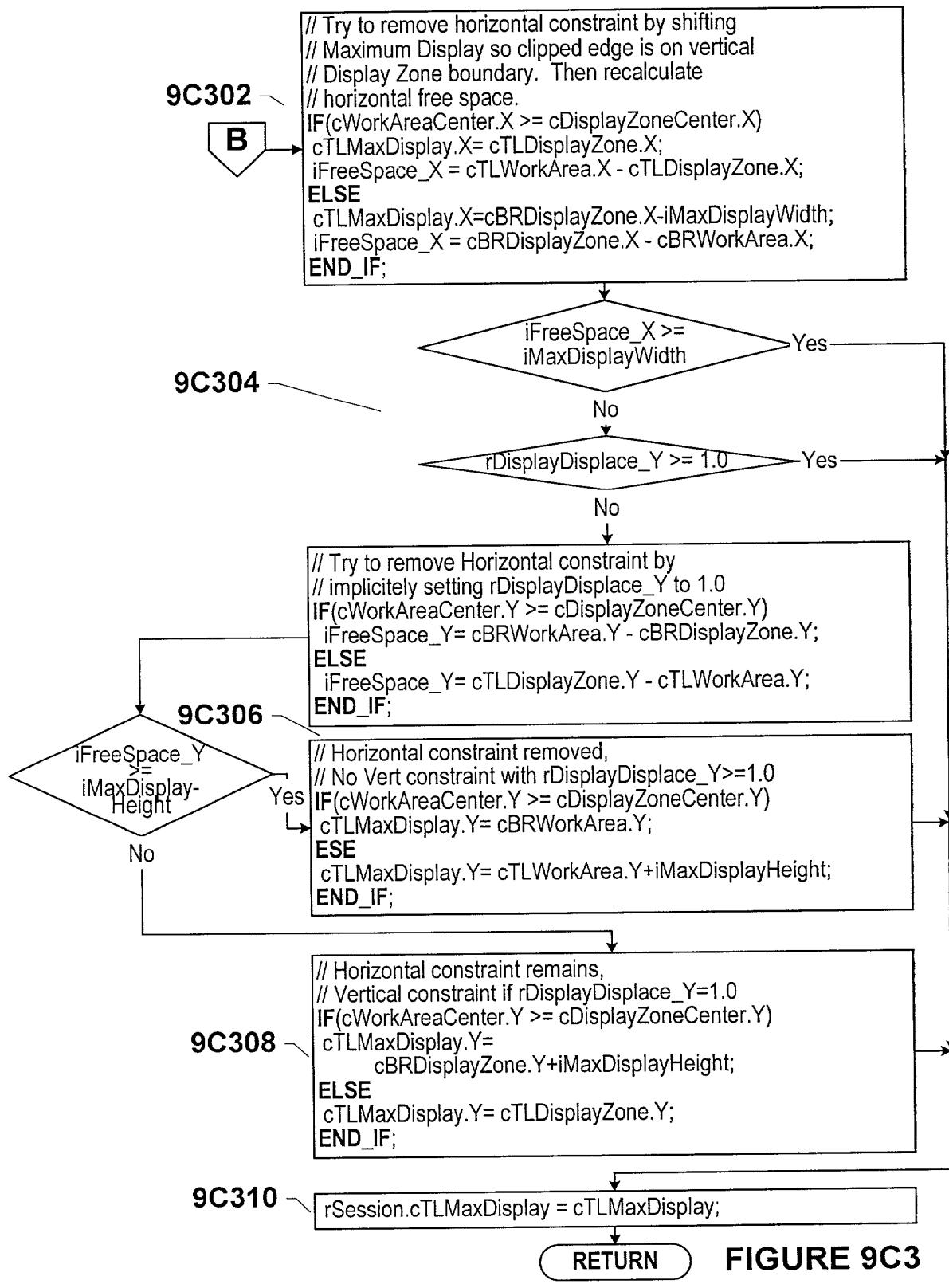
IF(iFreeSpace_Y < iMaxDisplayHeight) // Vertical constraint?
    bVert = TRUE;
ELSE
    bVert = FALSE;
END_IF;
```

9C204

```
// Optimum location for Maximum Display origin.
cWorkAreaCenter= rSession.cWorkAreaCenter;
cDisplayZoneCenter.X=(cTLDisplayZone.X+cBRDisplayZone.X)/2;
cDisplayZoneCenter.Y=(cTLDisplayZone.Y+cBRDisplayZone.Y)/2;
IF(cWorkAreaCenter.X >= cDisplayZoneCenter.X)
    cTLMaxDisplay.X = cOpt.X - iMaxDisplayWidth;
ELSE
    cTLMaxDisplay.X = cOpt.X;
END_IF

IF(cWorkAreaCenter.Y >= cDisplayZoneCenter.Y)
    cTLMaxDisplay.Y = cOpt.Y;
ELSE
    cTLMaxDisplay.Y = cOpt.Y + iMaxDisplayHeight;
END_IF;
```

9C206**FIGURE 9C2**



RETURN

FIGURE 9C3

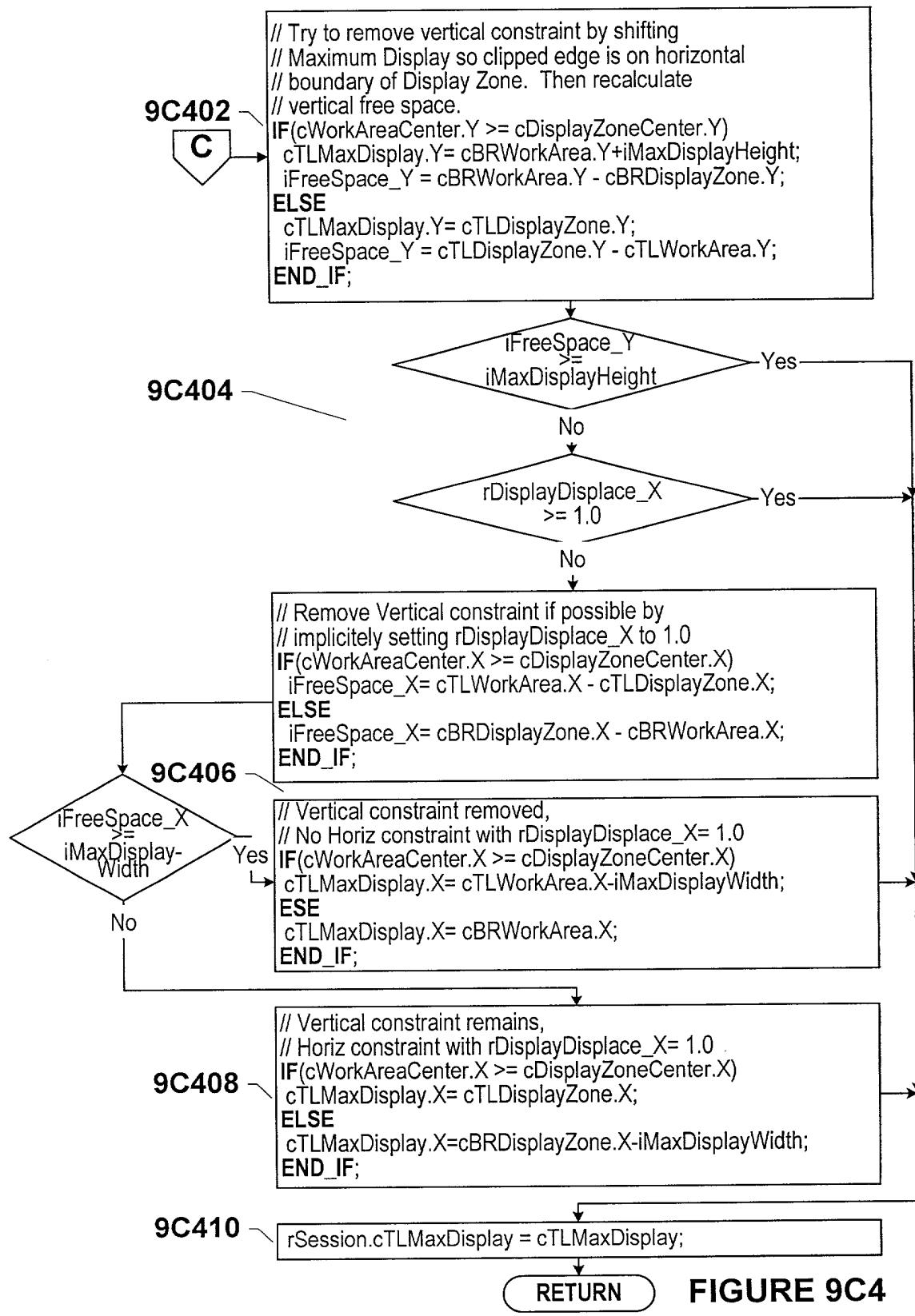


FIGURE 9C4

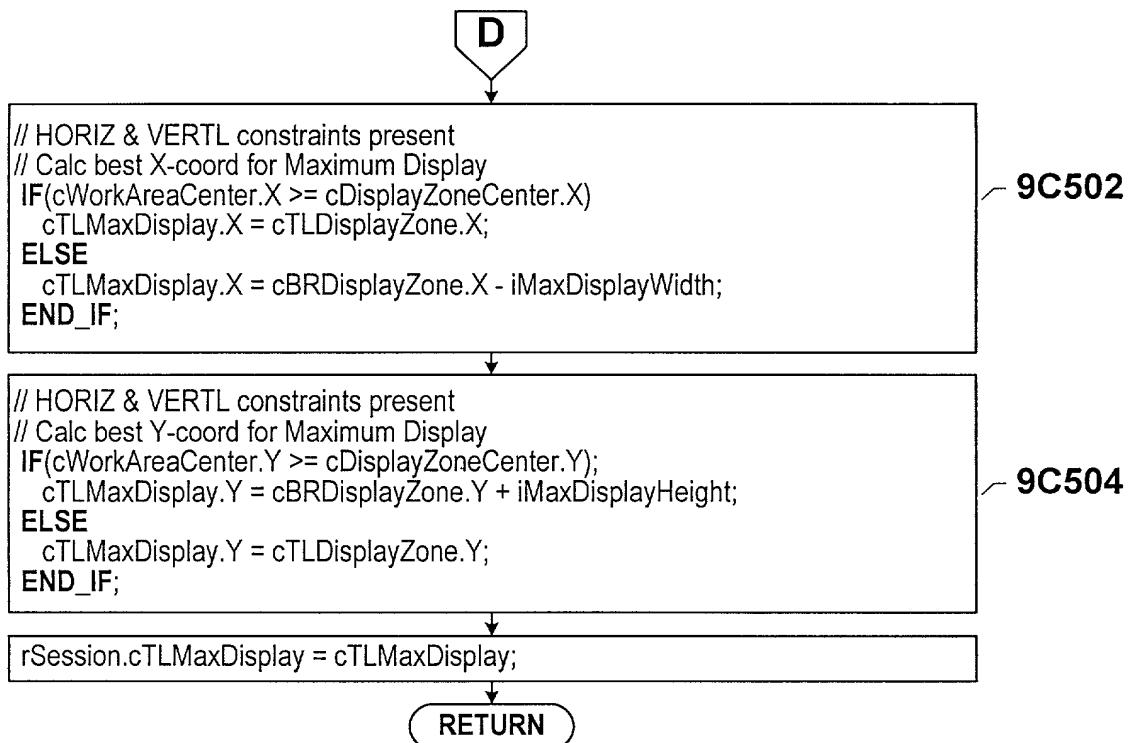


FIGURE 9C5

```
PROCEDURE CalculateFixedPointCoordinates(
    sControlType, rDisplayParms, rSession, llnCD);
```

9D00

```
// Fetch needed values from data structures
rFixedPointDisplace_X =
    rDisplayParms.aControl[sControlType].rVar.rFixedPointDisplace_X;
rFixedPointDisplace_Y =
    rDisplayParms.aControl[sControlType].rVar.rFixedPointDisplace_Y;
iMaxDisplayHeight= rSession.iMaxDisplayHeight;
iMaxShellWidth = rSession.iMaxShellWidth;
iMaxShellHeight = rSession.iMaxShellHeight;
cTLMaxDisplay = rSession.cTLMaxDisplay;
```

9D02

```
// Determine top-left corner of the maximum shell
cTLMaxShell.X= cTLMaxDisplay.X;
cTLMaxShell.Y= cTLMaxDisplay.Y-(iMaxDisplayHeight-iMaxShellHeight);
```

9D04

```
// Determine Fixed-Point Coordinates
pnCD = rSession.pnCD;
*pnCD.rCurrentDisplay.cFixedPoint.X =
    cTLMaxShell.X + (rFixedPointDisplace_X*iMaxShellWidth);
*pnCD.rCurrentDisplay.cFixedPoint.Y =
    cTLMaxShell.Y - (rFixedPointDisplace_Y*iMaxShellHeight);
```

9D06

```
// Translate axis to position the axis origin at the fixed-point
TRANSLATE_AXIS(FROM = rSessionParms.cInitialAxisOrigin,
                TO =*pnCD.rCurrentDisplay.cFixedPoint);
```

9D08

```
// Determine defining corners of display zone under translated axis
cTLDisplayZone = GET_DISPLAY_ZONE_ORIGIN();
iDisplayZoneWidth= GET_DISPLAY_ZONE_WIDTH();
iDisplayZoneHeight= GET_DISPLAY_ZONE_HEIGHT();
cBRDisplayZone.X= cTLDisplayZone.X + iDisplayZoneWidth;
cBRDisplayZone.Y= cTLDisplayZone.Y - iDisplayZoneHeight;
```

RETURN

FIGURE 9D

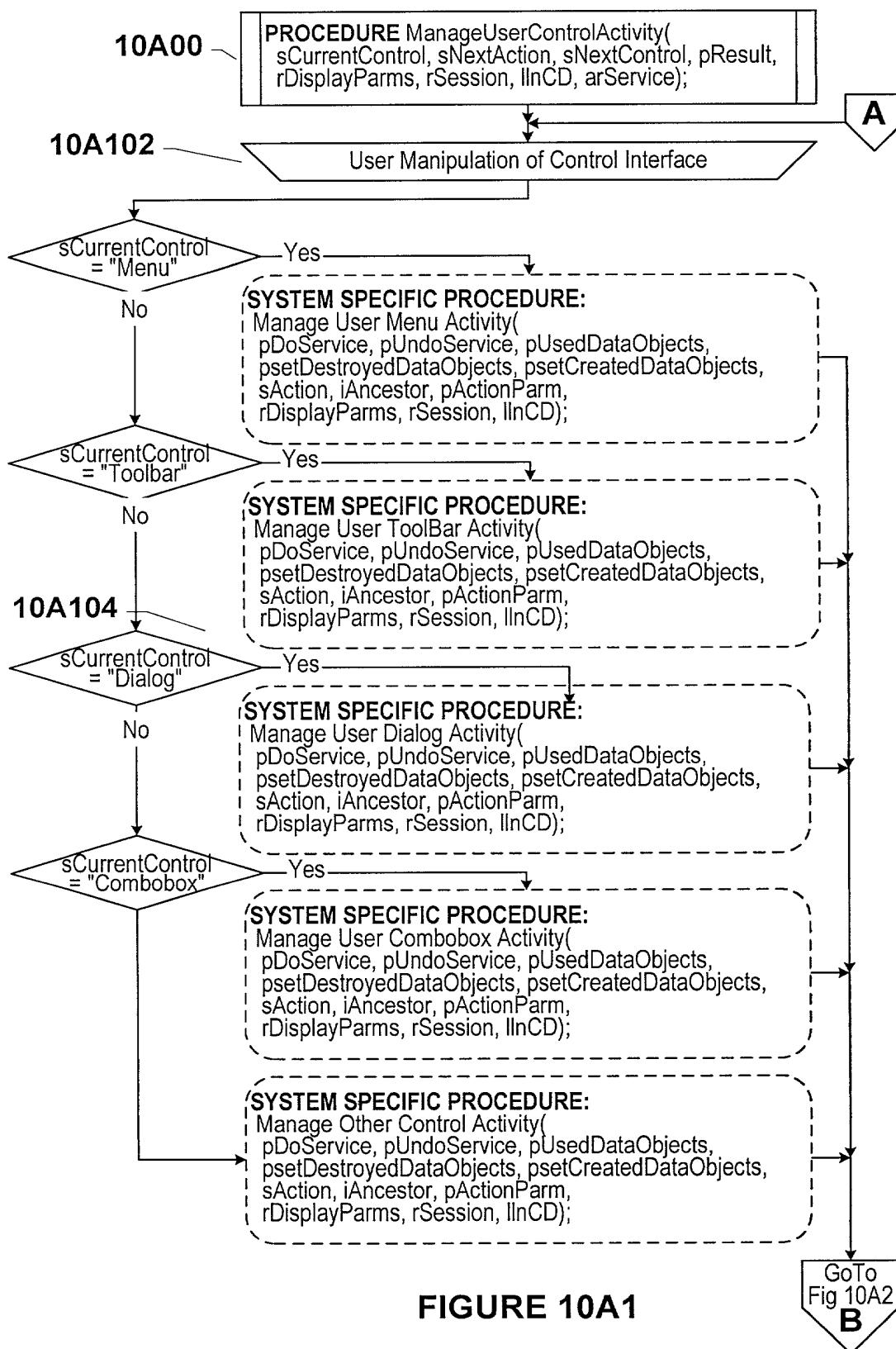


FIGURE 10A1

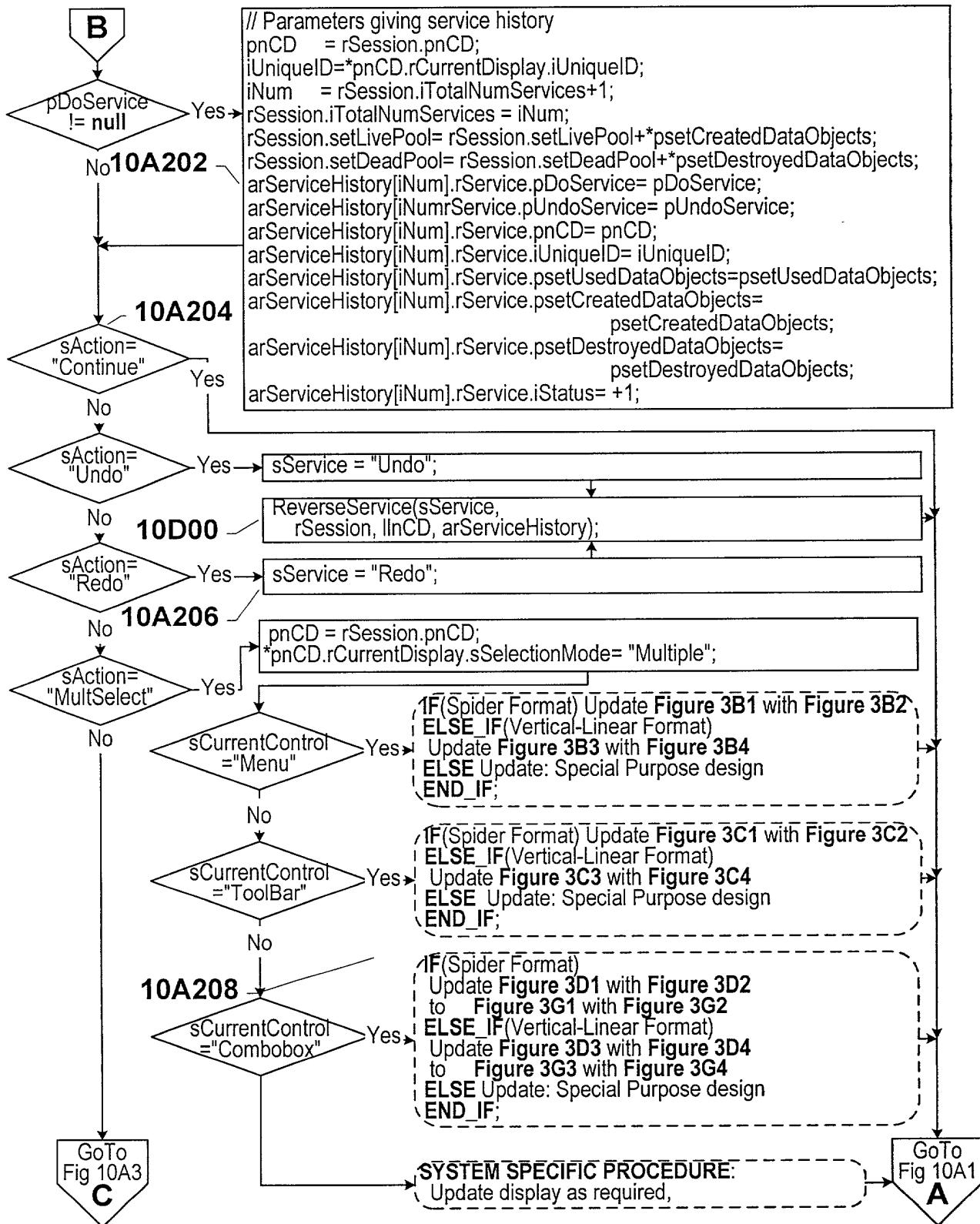


FIGURE 10A2

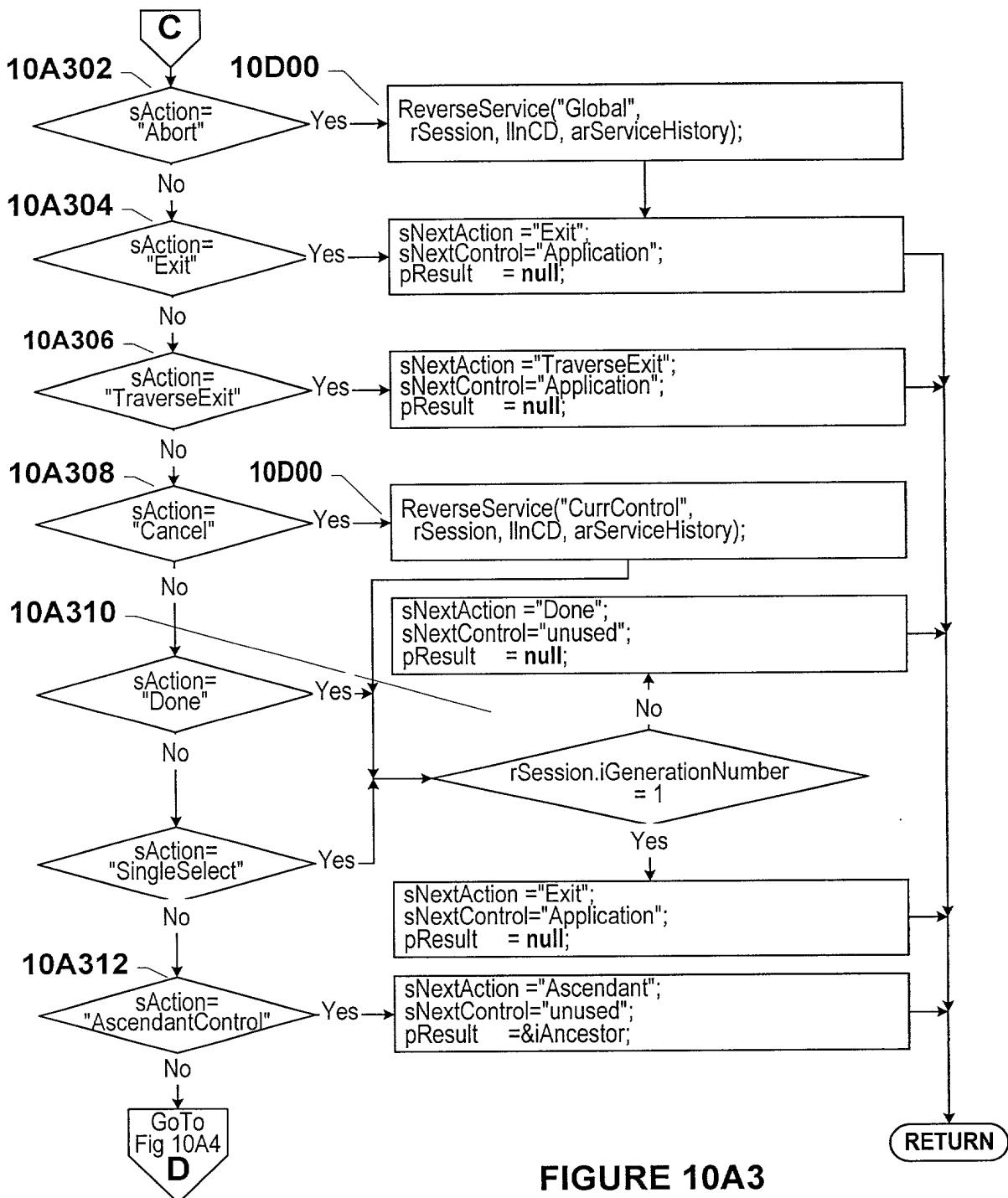


FIGURE 10A3

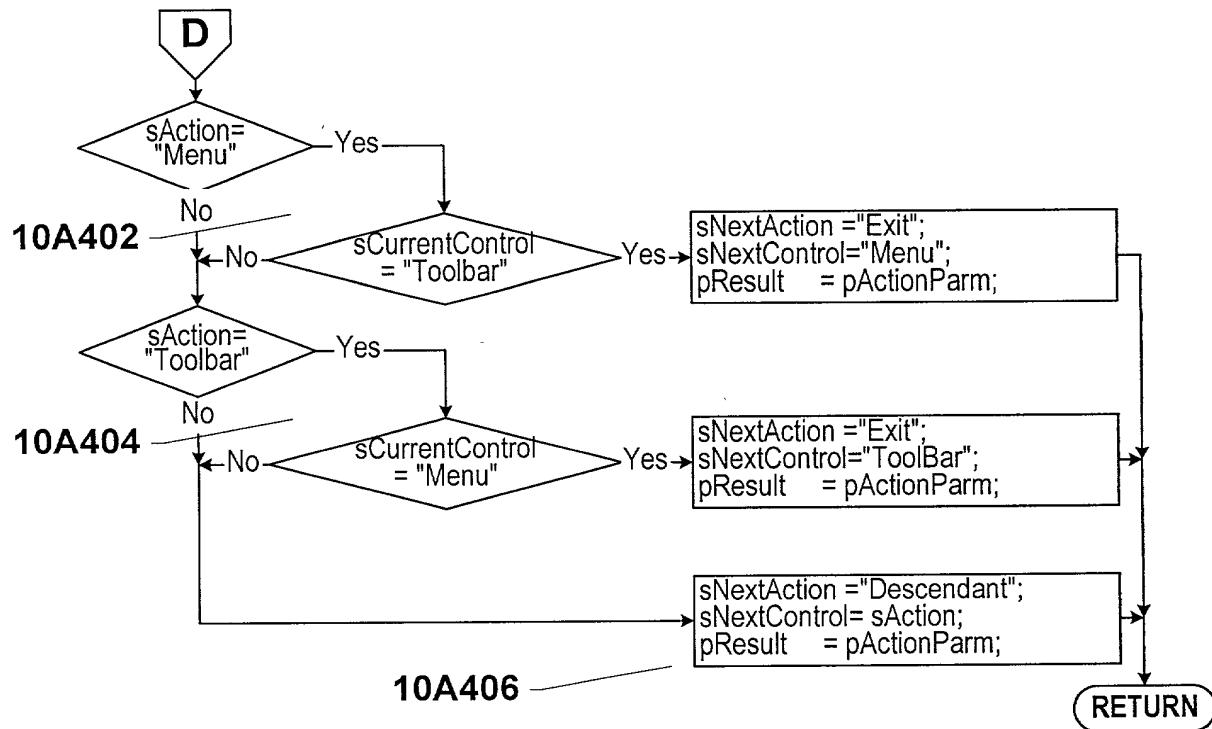
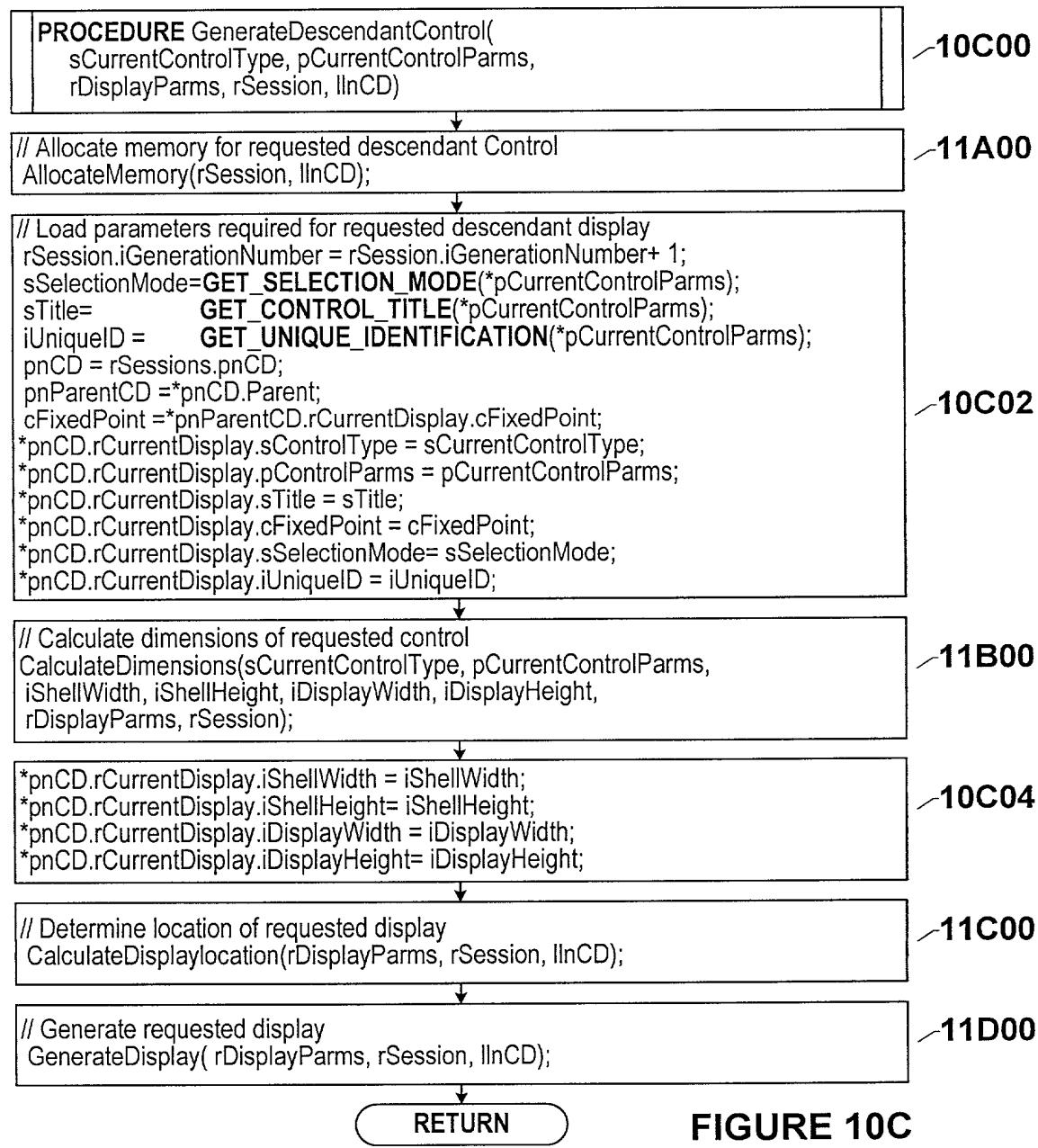
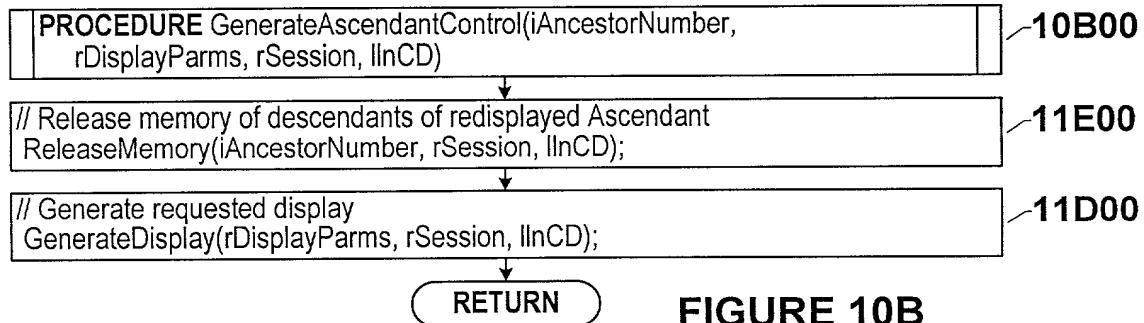


FIGURE 10A4



PROCEDURE ReverseService(sService,
rSession, llnAC, arService);

10D00

// NOTE: General

// A service may create or destroy data objects or alter their attributes.

// The following are presumed:

// "A" and "B" are services with A performed prior to B.

// "X" is a data object.

// NOTE: Create Data Object

// Data objects created by a service are destroyed if the service is reversed; i.e., "undone".

// Presume A then B are "undone" with the consequence that data object X is destroyed. If B

// references X, B cannot be "redone" after being "undone" unless A is first "redone".

// NOTE: Destroy Data Object

// Data objects destroyed by a service are returned to existence if the service is "undone".

// Presume A then B are "undone" which recreates X. Since X does not exist when B is initially

// performed, X is not referenced by B. An A "redo" destroying X is thus inconsequential to B.

// NOTE: Alter Data Object Attribute

// A service may reference a data object that has had attributes altered by a prior service. Since

// altering attributes neither creates nor destroys the data object, B can be redone irrespective

// of whether A is "redone". It is presumed that the user implementing A & B to alter an X

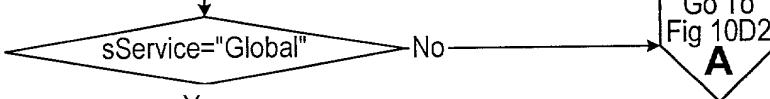
// attribute will be aware of the influence on B of "redoing" or not "redoing" A.

```

pnCD      = rSession.pnCD;
iUCurrControlID=*pnCD.rCurrentDisplay.iUniqueID;
iNServices = rSession.iTotalNumServices;
setInitialDataObjects= rSession.setInitialDataObjectsObjects;
setLivePool= rSession.setLivePool;
setDeadPool= rSession.setDeadPool;

```

10D102



// Undo all services performed during current
// activation of the control subsystem.
FOR i= iNServices DOWNTO 1 BY -1

```
iStatus= arServiceHistory.rServices[i].iStatus;
```

10D104

**SYSTEM SPECIFIC PROCEDURE:**

Undo service pointed to by
arServiceHistory.rServices[i].pUndoService

END(Global Undo)

RETURN

FIGURE 10D1

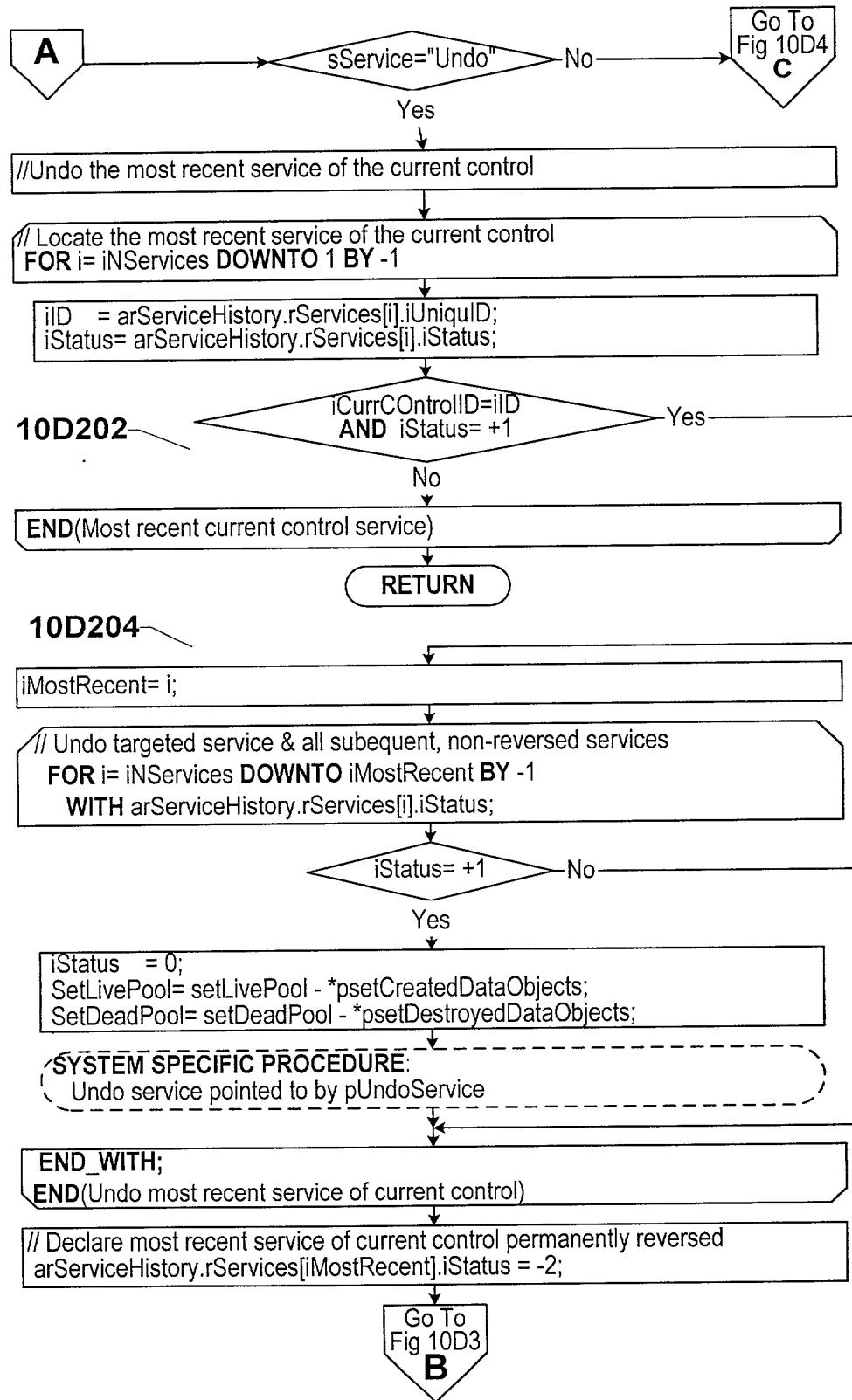
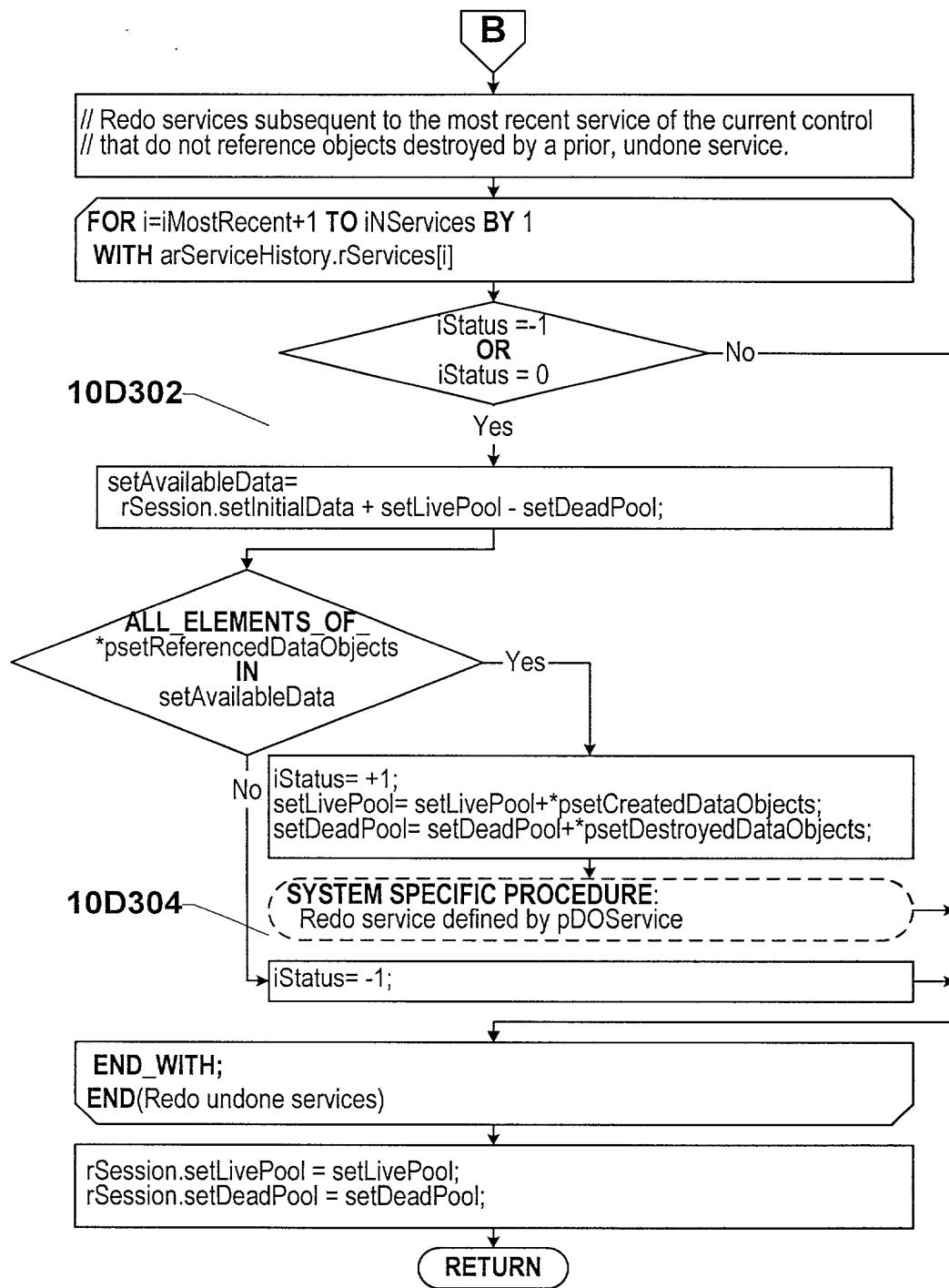


FIGURE 10D2

**FIGURE 10D3**

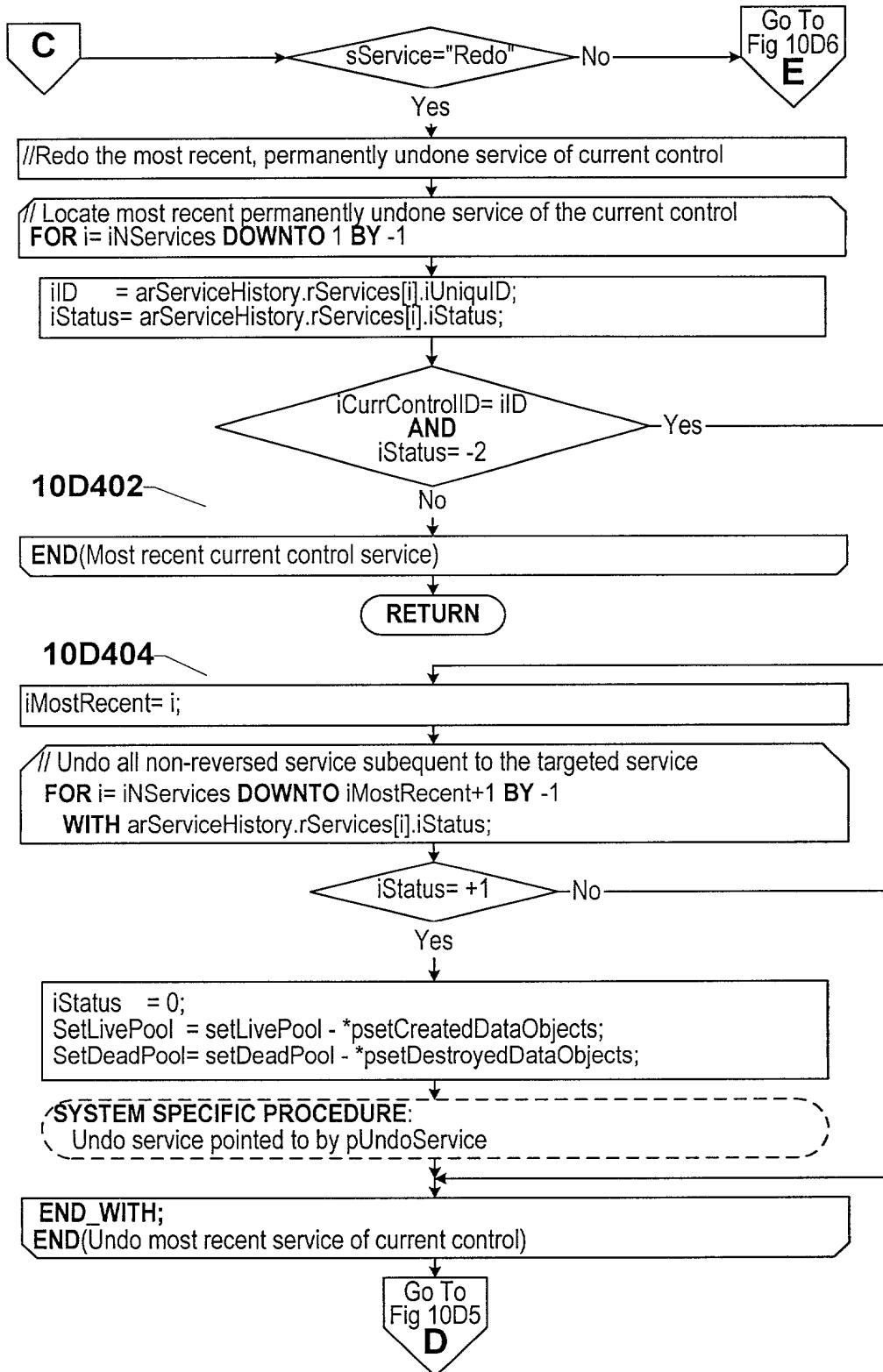


FIGURE 10D4

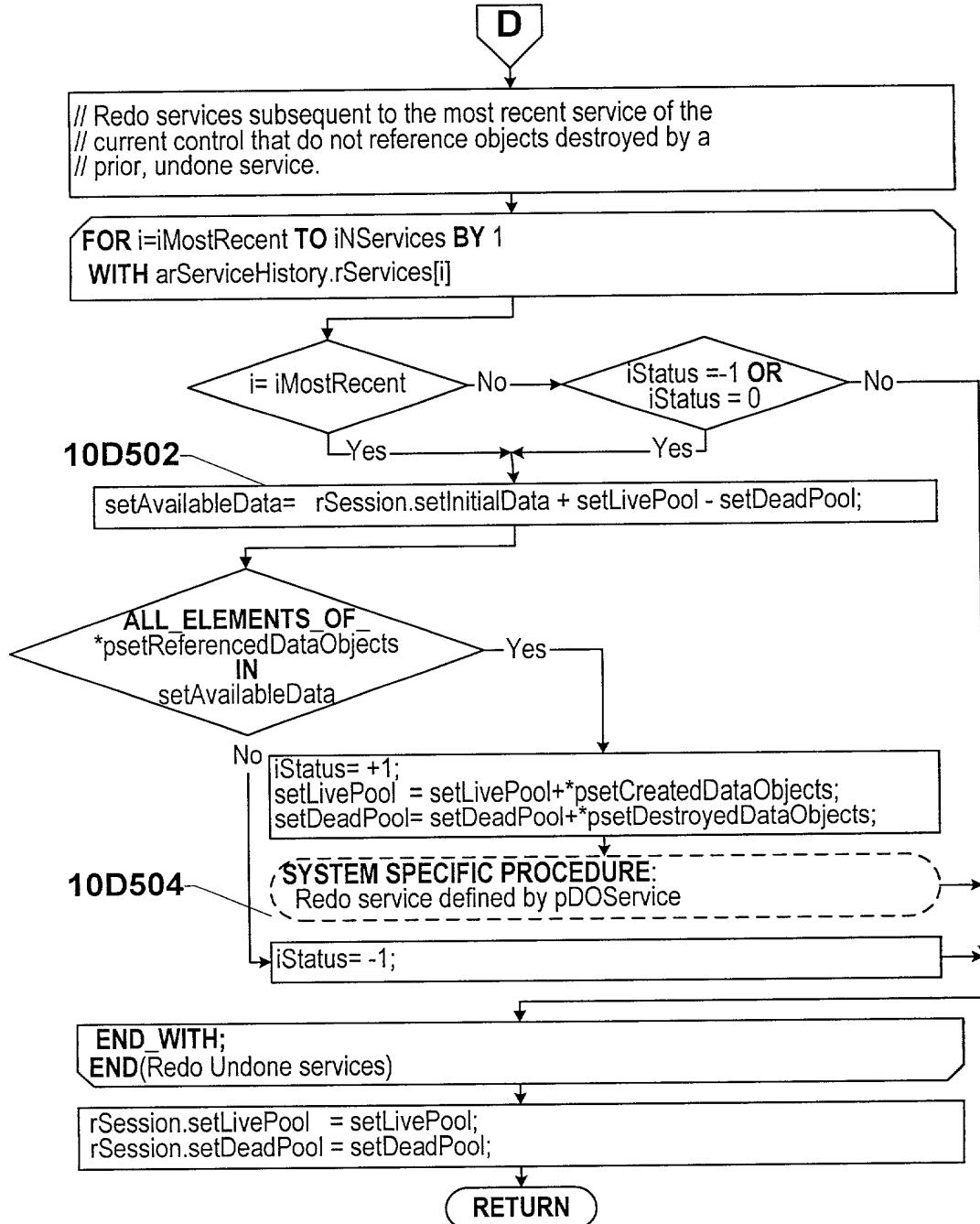


FIGURE 10D5

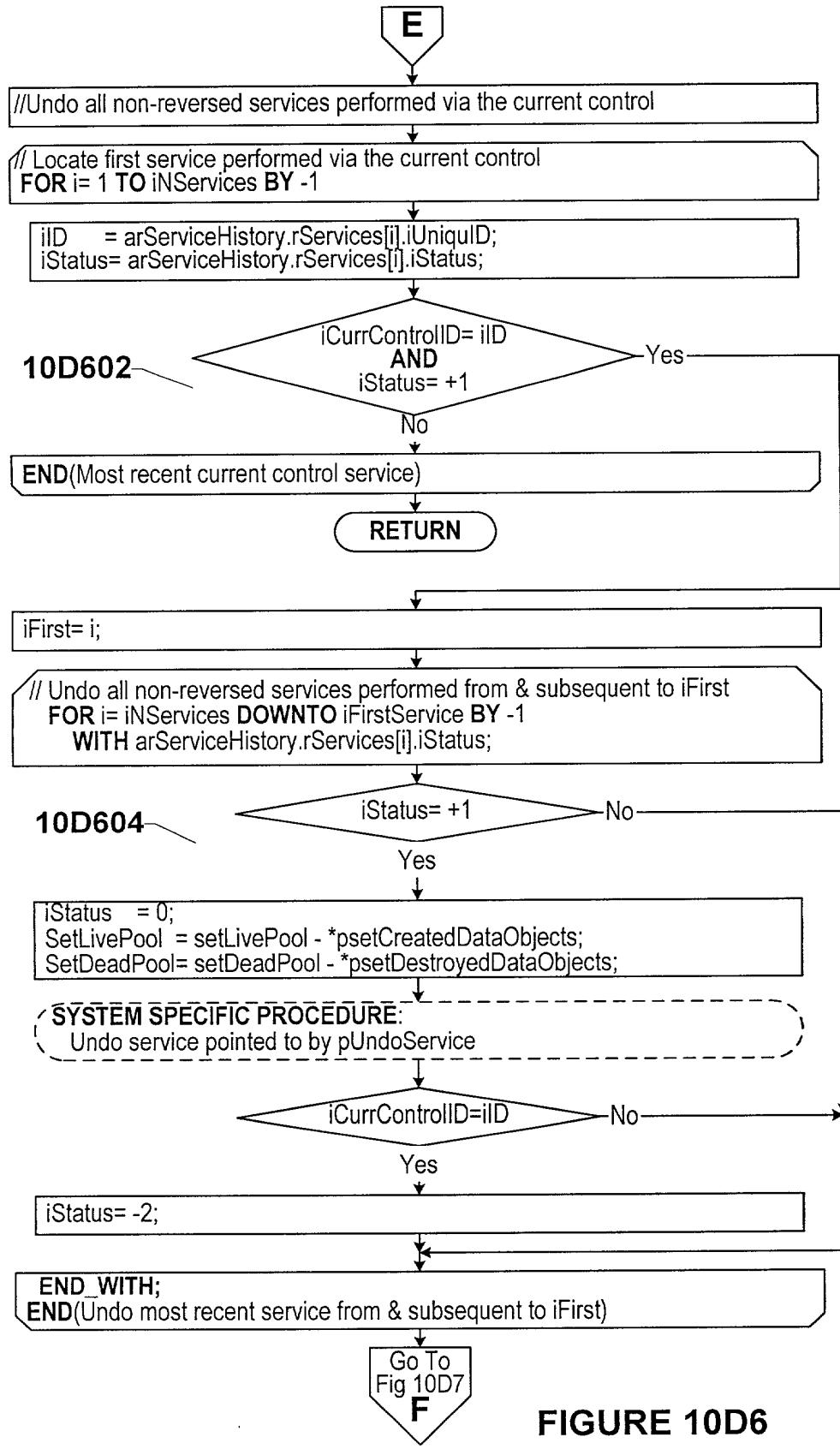


FIGURE 10D6

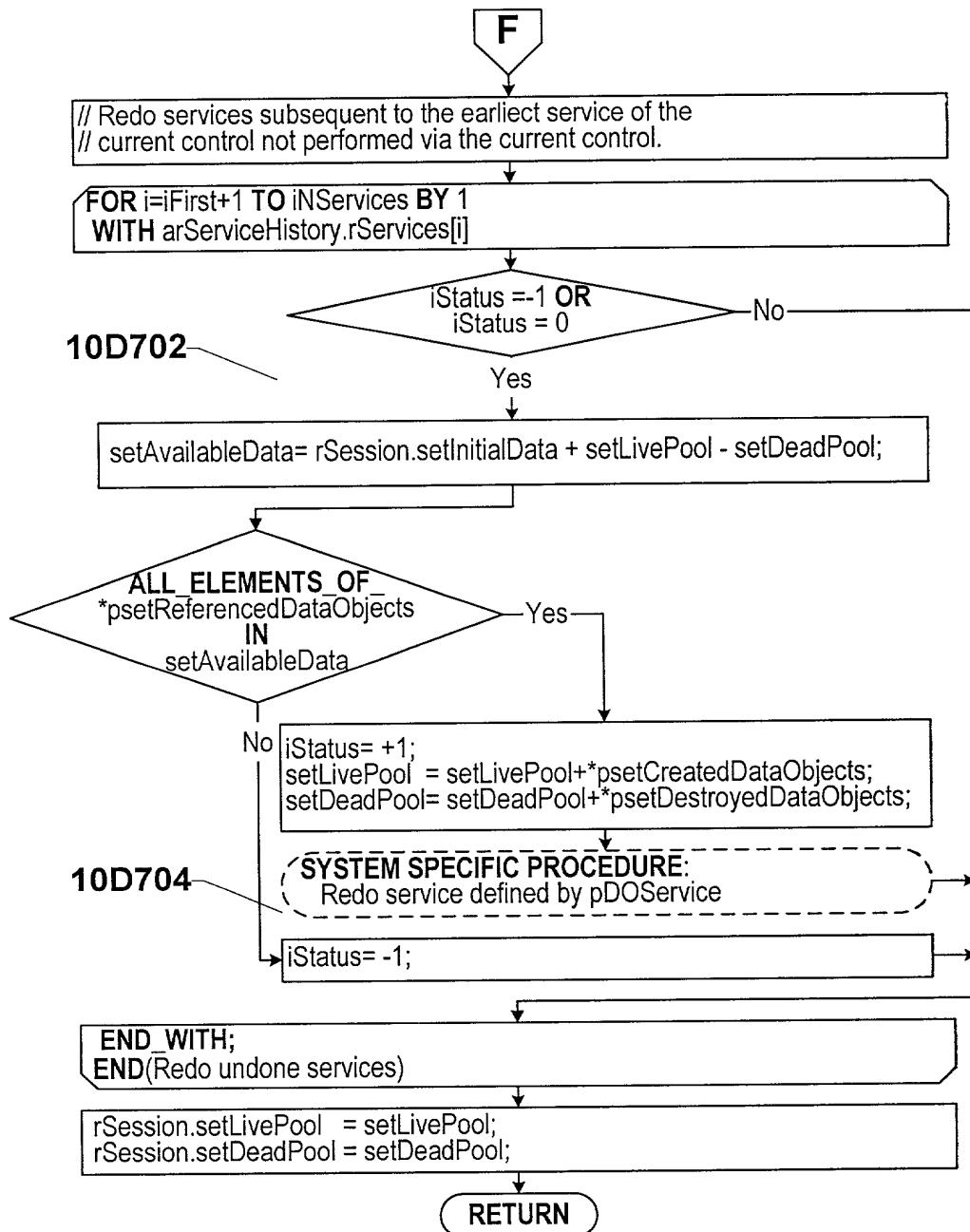


FIGURE 10D7

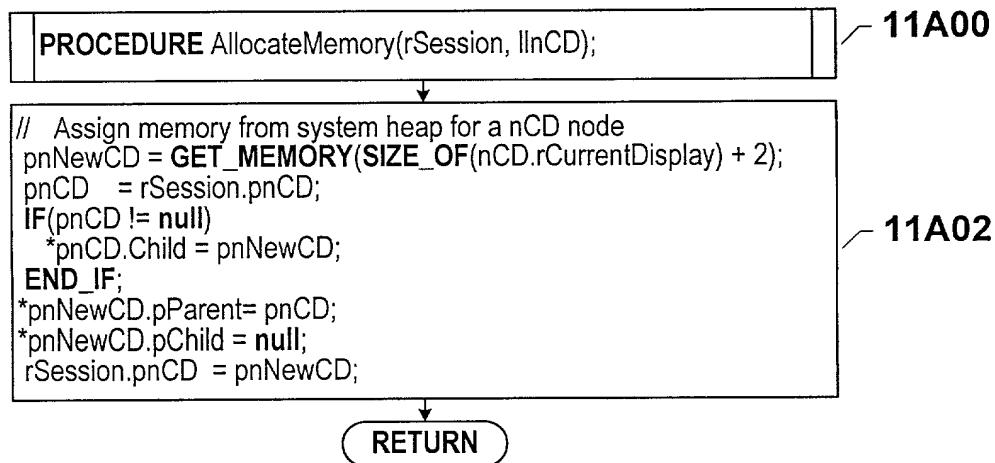


FIGURE 11A

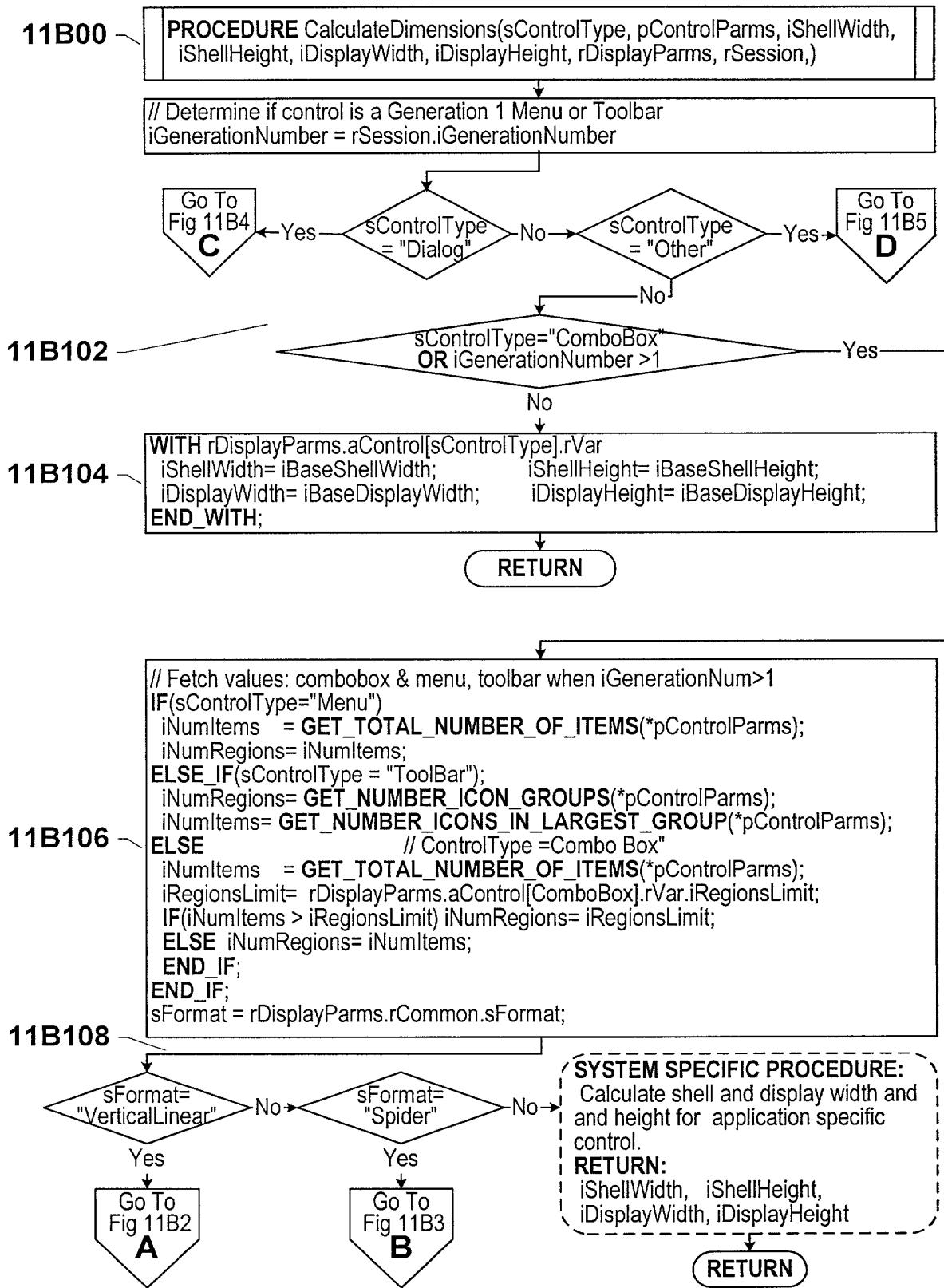


FIGURE 11B1

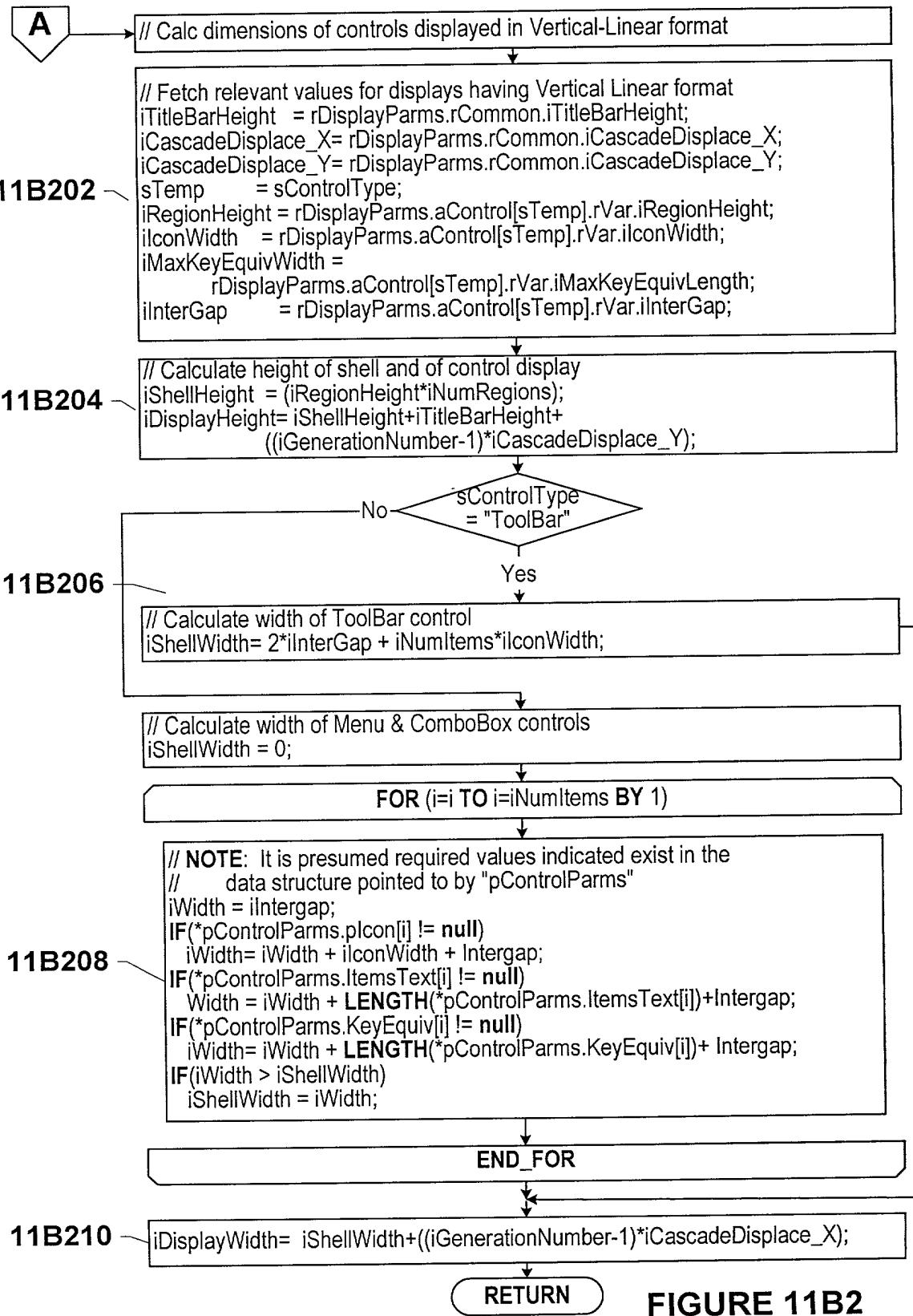
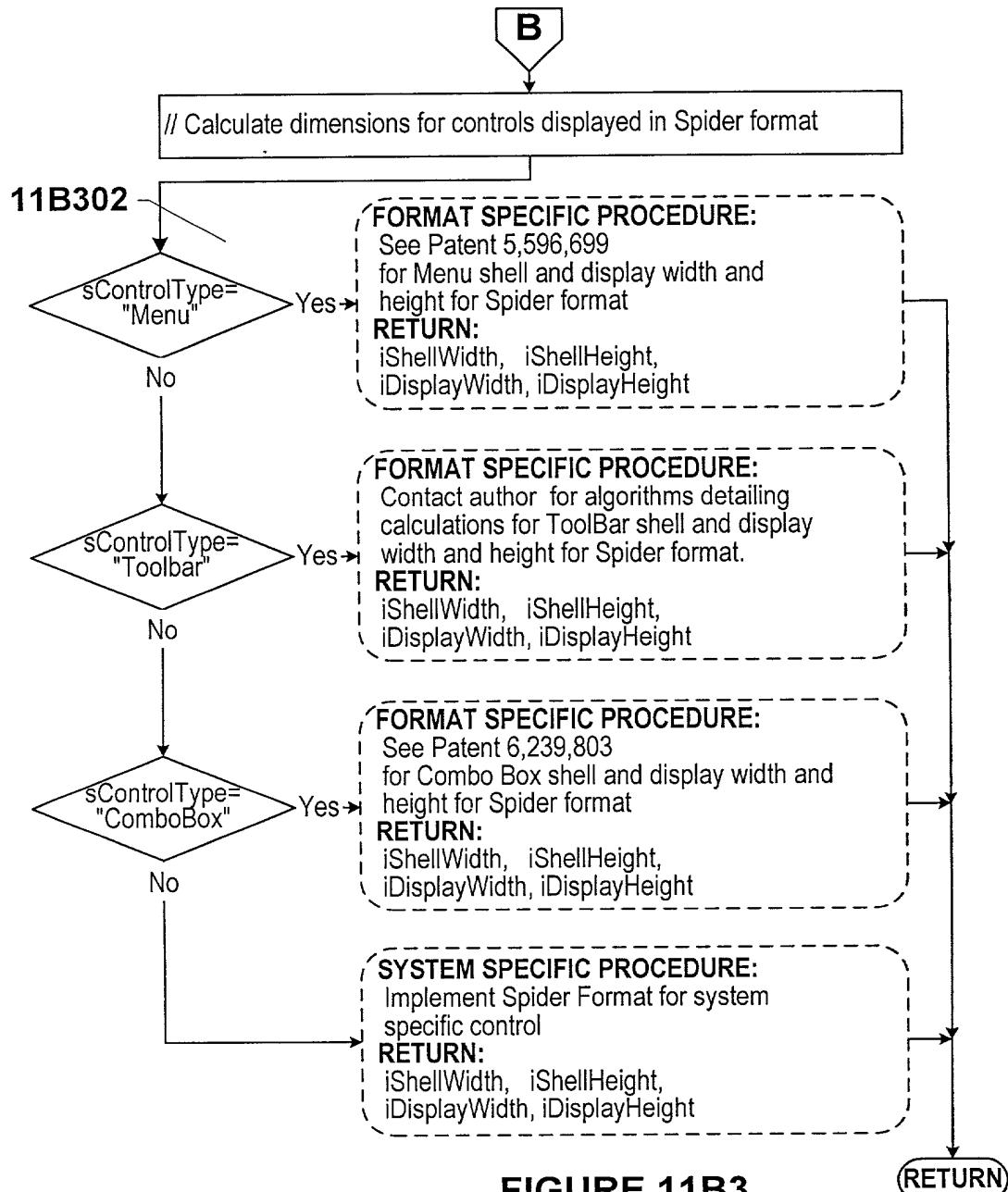


FIGURE 11B2

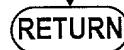


11B402



```
// Determine Dialog Box Width
iShellWidth = GET_CONTROL_WIDTH(*pControlParms);
iDisplayWidth =
    iShellWidth + (iGenerationNumber-1)*iCascadeDisplace_X;
```

```
// Determine Dialog Box Height
iDisplayHeight= GET_CONTROL_HEIGHT(*pControlParms);
sTitle = GET_CONTROL_TITLE(*pControlParms);
IF(sTitle = null)
    iShellHeight= iDisplayHeight;
ELSE
    iShellHeight= iDisplayHeight - iTitleBarHeight;
END_IF;
iDisplayHeight =
    iDisplayHeight + (iGenerationNumber-1)*iCascadeDisplace_Y;
```

**FIGURE 11B4**

11B502

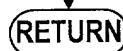
**SYSTEM SPECIFIC PROCEDURES:**

| Generate values of maximum width and height for "Other" controls

| presented via pop-up display.

|**RETURN:**

| iShellWidth, iShellHeight, iDisplayWidth, iDisplayHeight

**FIGURE 11B5**

PROCEDURE CalculateDisplayLocation(rDisplayParms, rSession, llnCD);

11C00

```
// Extract required values from data structures
pnCD = rSession.pnCD;
iGenNo = rSession.iGenerationNumber;
sControlType = *pnCD.rCurrentDisplay.sControlType;
iShellWidth = *pnCD.rCurrentDisplay.iShellWidth;
iShellHeight = *pnCD.rCurrentDisplay.iShellHeight;
iTitleBarHeight = rDisplayParms.rCommon.iTitleBarHeight;
iCascadeDisplace_X = rDisplayParms.rCommon.iCascadeDisplace_X;
iCascadeDisplace_Y = rDisplayParms.rCommon.iCascadeDisplace_Y;
rFixedPointDisplace_X = rDisplay.aControl[sControlType].rVar.cFixedPointDisplace_X;
rFixedPointDisplace_Y = rDisplay.aControl[sControlType].rVar.cFixedPointDisplace_Y;
```

11C02

```
// Determine cTLShell NOTE: Axis origin translated to fixed-point coords by 9D08
cTLShell.X = -(rFixedPointDisplace_X*iShellWidth);
cTLShell.Y = +(rFixedPointDisplace_Y*iShellHeight);
*pnCD.rCurrentDisplay.cTLShell.X = cTLShell.X;
*pnCD.rCurrentDisplay.cTLShell.Y = cTLShell.Y;
```

11C04

```
// Determine coords of top-left bottom-right corners of display
cTLDisplay.X = cTLShell.X;
cTLDisplayWindow.Y = iTitleBarHeight+cTLShell.Y+((iGenNo-1)*iCascadeDisplace_Y);
cBRDisplay.X = cTLDisplay.X +(iShellWidth+((iGenNo-1)*iCascadeDisplace_X));
cBRDisplay.Y = cTLDisplay.Y - (iLabelHeight+iShellHeight+((iGenNo-1)*iCascadeDisplace_Y));
*pnCD.rCurrentDisplay.cTLDisplayWindow= cTLDisplay;
*pnCD.rCurrentDisplay.cBRDisplayWindow= cBRDisplay;
```

```
// Test if large display extends beyond display zone.
cTLDisplayZone = GET_DISPLAY_ZONE_ORIGIN();
iDisplayZoneWidth = GET_DISPLAY_ZONE_WIDTH();
iDisplayZoneHeight = GET_DISPLAY_ZONE_HEIGHT();
cBRDisplayZone.X = cTLDisplayZone.X + iDisplayZoneWidth;
cBRDisplayZone.Y = cTLDisplayZone.Y - iDisplayZoneHeight;
IF(cTLDisplayZone.X > cTLDisplay.X) iShift_X= cTLDisplayZone.X - cTLDisplay.X;
ELSEIF(cBRDisplayZone.X < cBRDisplay.X) iShift_X= cBRDisplayZone.X-cBRDisplay.X;
ELSE iShift_X = 0; END_IF
IF(cTLDisplayZone.Y > cTLDisplay.Y) iShift_Y= cTLDisplayZone.Y-cTLDisplay.Y;
ELSEIF(cBRDisplayZone.Y < cBRDisplay.Y) iShift_Y= cBRDisplayZone.Y-cBRDisplay.Y;
ELSE iShift_Y = 0; END_IF;
```

11C06

```
// Adjust fixed point as required to avoid clipping display
IF(iShift_X != 0 OR iShift_Y != 0)
WITH *pnCD.rCurrentDisplay
TRANSLATE_AXIS(  

    FROM=cFixedPoint TO=rSessionParms.iInitialAxisOrigin);
cFixedPoint.X = cFixedPoint.X + iShift_X;
cFixedPoint.Y = cFixedPoint.Y + iShift_Y;
TRANSLATE_AXIS(  

    FROM=rSessionParms.iInitialAxisOrigin TO=cFixedPoint);
END_WITH; END_IF;
```

11C08

RETURN

FIGURE 11C

PROCEDURE GenerateDisplay(rDisplayParms, rSession, llnCD);

- 11D00

```
iTitleBarHeight = rDisplayParms.rCommon.iTitleBarHeight;
iCascadeDisplace_X= rDisplayParms.rCommon.iCascadeDisplace_X;
iCascadeDisplace_Y= rDisplayParms.rCommon.iCascadeDisplace_Y;
pCurrGhostNode = rSession.pnInitialCD
iGenerationNumber = rSession.iGenerationNumber;
```

```
pnCD = rSession.pnCD;
sControlType = *pnCD.rCurrentControl.sControlType;
pControlParms = *pnCD.rCurrentControl.pControlParms;
cTLShell = *pnCD.rCurrentDisplay.cTLShell;
iShellWidth = *pnCD.rCurrentDisplay.iShellWidth;
iShellHeight = *pnCD.rCurrentDisplay.iShellHeight;
iDisplayWidth = *pnCD.rCurrentDisplay.iDisplayWidth;
iDisplayHeight = *pnCD.rCurrentDisplay.iDisplayHeight;
cTLDISPLAYWindow = *pnCD.rCurrentDisplay.cTLDISPLAYWindow;
```

- 11D102

```
ALTER_WINDOW(HANDLE=rSession.handleDisplayWindow,
RENDER = YES,
ORIGIN = cTLDISPLAYWindow,
WIDTH = iDisplayWidth, HEIGHT = iDisplayHeight);
```

- 11D104

FOR i= 1 **TO** (iGenerationNumber-1) **BY** 1

- 11D106

```
// Determine origin of current ghost display shell
iNumOffset = iGenerationNumber - i;
cTLDISPLAY.X= cTLShell.X + iNumOffset*iCascadeDisplace_X;
cTLDISPLAY.Y= iTitleBarHeight+cTLShell.Y+iNumOffset*iCascadeDisplace_Y;
```

- 11D108

```
// Generate ghost display
RECTANGLE(
ORIGIN= (cTLDISPLAY.X, (cTLDISPLAY.Y-iTitleBarHeight)),
WIDTH = iShellWidth, HEIGHT= iShellHeight,
TEXT = null, FILL = <light gray>);
```

- 11D110

```
// Generate title bar for ghost display
RECTANGLE(
ORIGIN=(cTLDISPLAY.X, cTLDISPLAY.Y),
WIDTH = iShellWidth, HEIGHT= iTitleBarHeight,
TEXT(LEFT_JUSTIFY) = *pCurrGhostNode.sTitle, FILL = <label fill color>);
```

- 11D112

```
// Fetch address of next ghost display
pCurrGhostNode = *pCurrGhostNode.pChild;
```

- 11D114

END_FOR



FIGURE 11D1

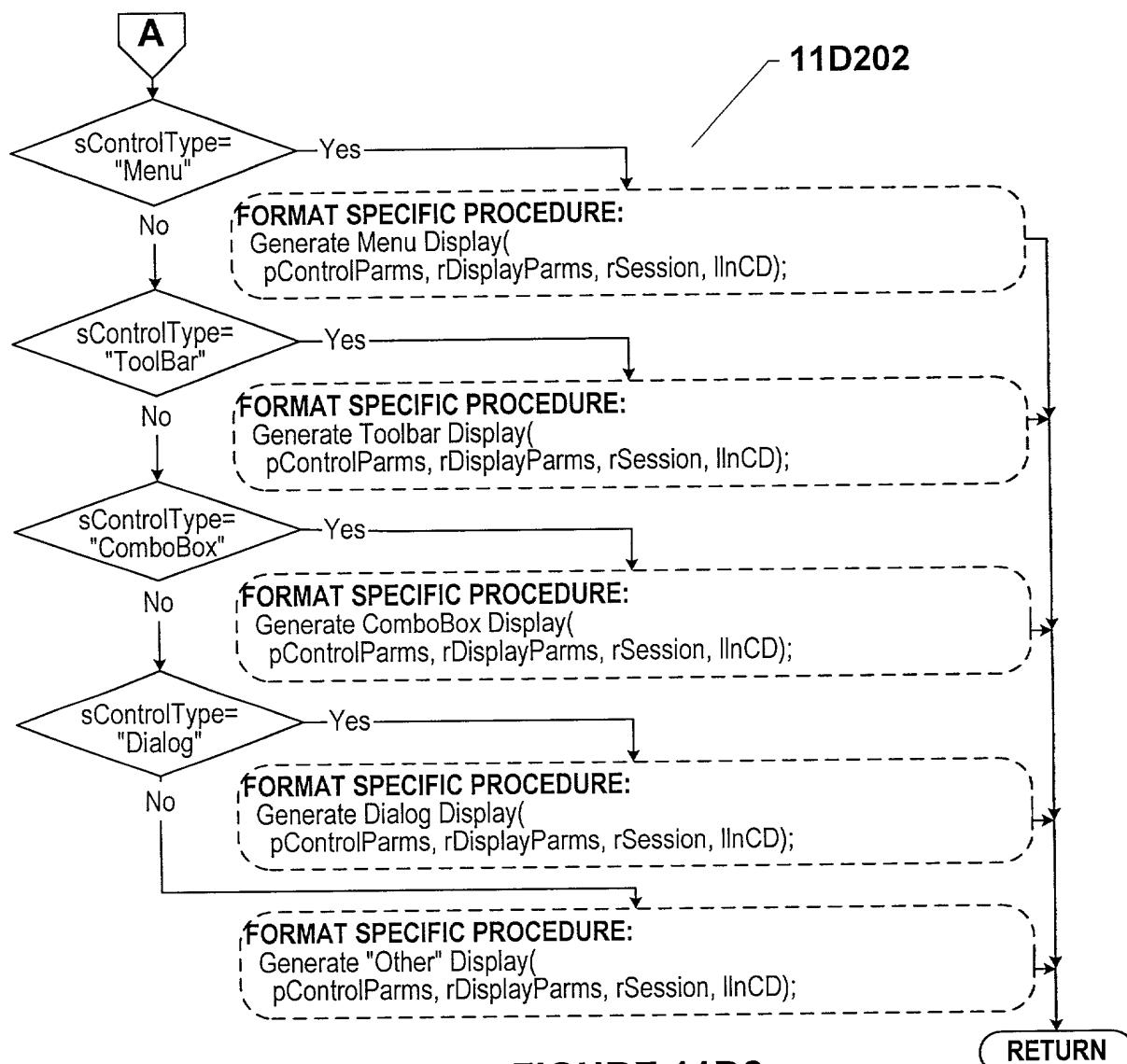
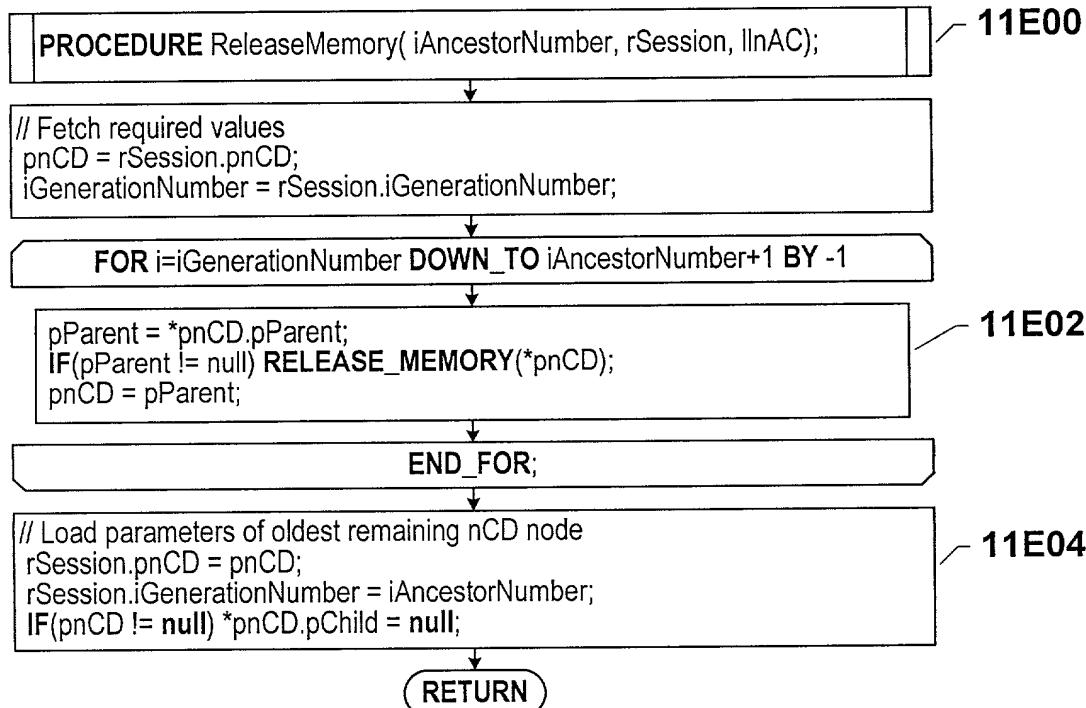
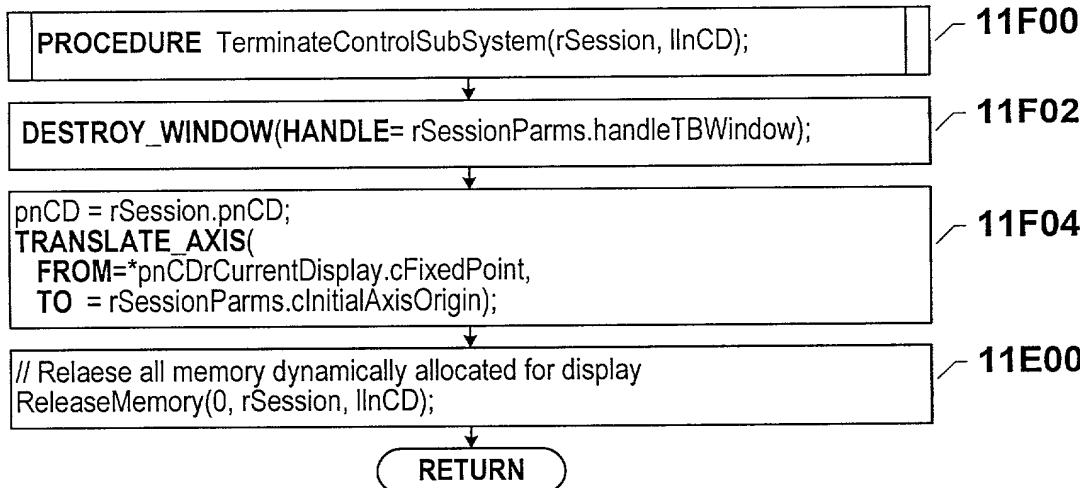


FIGURE 11D2

**FIGURE 11E****FIGURE 11F**